AWS Marketplace
Seller Guide
# Table of Contents

- **What Is AWS Marketplace?** ................................................................. 1  
  Using AWS Marketplace as a seller ......................................................... 1
- **Getting started as a seller** ................................................................. 3  
  Seller requirements for publishing free software products .................. 3  
  Additional seller requirements for paid products ................................. 3  
  Eligible jurisdictions for paid products .................................................. 4
- **AWS Marketplace Management Portal** .......................................... 4
  Seller registration process ..................................................................... 5  
  US bank account for sellers of paid products ....................................... 5
  AWS Marketplace Tax Calculation Service ......................................... 6
  Already a seller? ................................................................................... 7
- **Seller toolkit** .................................................................................... 7
  AWS Marketplace Commerce Analytics Service .................................. 7
  Enhanced Data Sharing Program ......................................................... 18
  AWS Marketplace Field Demonstration Program ............................... 19
  Product Support Connection ............................................................... 19
- **Preparing your product** .................................................................... 29
  Product pricing ..................................................................................... 30  
  AWS charges versus software charges ................................................. 30  
  Annual pricing .................................................................................... 31  
  Usage pricing ...................................................................................... 32  
  Private offers ...................................................................................... 32  
  SaaS subscriptions pricing .................................................................. 32  
  SaaS contracts pricing ........................................................................ 33  
  AMI pricing models ............................................................................ 33  
  Pricing your software with SaaS .......................................................... 35
  Annual products .................................................................................. 40
  Refunds ............................................................................................... 41
- **Regions and countries** ...................................................................... 43  
  AWS Regions ...................................................................................... 43
  Countries ............................................................................................. 43
- **Private offers** .................................................................................. 43  
  How private offers work ...................................................................... 44  
  Private offers through consulting partners ......................................... 44  
  Notes about private offers ................................................................... 44
  Supported product types ....................................................................... 45  
  Offer submission process ................................................................... 45
  Reporting for private offers ................................................................. 46
  Flexible payment scheduler ................................................................. 46
  Consulting partner private offers ......................................................... 47
  Private offer upgrades and renewals ..................................................... 50
- **Standardized license terms** .............................................................. 52
  Standard Contract for AWS Marketplace (SCMP) .............................. 52
  Enterprise Contract for AWS Marketplace (ECMP) ............................ 53
- **Categories and metadata** ................................................................. 54
  Naming and describing your product .................................................. 54
  Choosing categories and keywords ..................................................... 55
- **Search engine optimization** .............................................................. 56
  Keywords ........................................................................................... 56
  Software categories ............................................................................ 56
  Highlights section .............................................................................. 57
  Short description ................................................................................ 57
  Example ............................................................................................. 57
- **AWS Marketplace for Desktop Applications (AMDA)** ...................... 58
Starting the onboarding process ................................................................. 58
Product submission and packaging ......................................................... 58
Application packaging types ................................................................. 59
Building the AMDA package ................................................................. 59
Application metadata .............................................................................. 60
Ingestion and new version updates ......................................................... 60
AMI-based products ................................................................................ 61
Multiple versions .................................................................................. 61
AMI file upload .................................................................................... 61
Removing products from AWS Marketplace ........................................... 61
Best practices for building AMIs ............................................................ 62
Rights ..................................................................................................... 62
Building an AMI .................................................................................. 62
Securing an AMI .................................................................................. 63
Metering service .................................................................................. 64
Metering service concepts .................................................................... 64
Pricing your software ............................................................................ 65
Adding your product to AWS Marketplace ............................................. 66
Modifying your software to use the Metering Service ......................... 67
Call AWS Marketplace Metering Service .............................................. 39
Failure handling .................................................................................. 39
Limitations ............................................................................................ 70
AMI-based Delivery Using CloudFormation .......................................... 70
Building your product listing ................................................................. 71
Preparing your AWS CloudFormation template .................................... 71
Getting the cost estimate for your template infrastructure .................... 72
Topology diagram ................................................................................ 73
Meeting the submission requirements .................................................... 73
Submitting your product request .......................................................... 74
Adding Serverless Application Components ........................................... 74
Private images ....................................................................................... 80
Package group requirements ................................................................. 81
Submitting your package group ............................................................. 81
Scan status ............................................................................................ 82
Submitting your product to AWS Marketplace ....................................... 83
AMI product checklist .......................................................................... 83
Container-based products .................................................................... 85
Getting help ......................................................................................... 85
Getting started with container products ............................................... 85
Creating a container product ................................................................. 85
Downloading and filling out the product load form for container products 86
Integrating metering for your container product .................................... 87
Publishing container products ............................................................... 87
Container product scans ....................................................................... 88
Container-based product requirements .................................................. 88
Security requirements .......................................................................... 88
Access requirements ............................................................................ 89
Customer information requirements ...................................................... 89
Product usage requirements ................................................................. 89
Architecture requirements ..................................................................... 89
Pricing container products .................................................................... 90
Pricing models for container products .................................................... 90
AWS Marketplace Metering Service integration .................................... 91
Entitlement ............................................................................................ 91
Integration guidelines ........................................................................... 92
Supported AWS Regions ....................................................................... 92
Preventing metering modification ......................................................... 92
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine learning products</td>
<td>106</td>
</tr>
<tr>
<td>Getting started with Amazon SageMaker</td>
<td>106</td>
</tr>
<tr>
<td>Amazon SageMaker algorithms and model packages</td>
<td>106</td>
</tr>
<tr>
<td>Algorithms and models</td>
<td>107</td>
</tr>
<tr>
<td>Package your code using Docker</td>
<td>107</td>
</tr>
<tr>
<td>Create your algorithm in SageMaker</td>
<td>108</td>
</tr>
<tr>
<td>Create your model package in SageMaker</td>
<td>109</td>
</tr>
<tr>
<td>Add your algorithm or model package to AWS Marketplace</td>
<td>110</td>
</tr>
<tr>
<td>Monetize your algorithm or model package</td>
<td>111</td>
</tr>
<tr>
<td>Best practices for samples</td>
<td>111</td>
</tr>
<tr>
<td>Software as a service (SaaS)–based products</td>
<td>113</td>
</tr>
<tr>
<td>Getting started</td>
<td>113</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>113</td>
</tr>
<tr>
<td>SaaS subscriptions</td>
<td>114</td>
</tr>
<tr>
<td>SaaS contracts</td>
<td>116</td>
</tr>
<tr>
<td>SaaS contracts with consumption</td>
<td>117</td>
</tr>
<tr>
<td>Plan your SaaS product</td>
<td>119</td>
</tr>
<tr>
<td>Plan your pricing</td>
<td>120</td>
</tr>
<tr>
<td>Plan your billing integration</td>
<td>120</td>
</tr>
<tr>
<td>Plan your Amazon SNS integration</td>
<td>120</td>
</tr>
<tr>
<td>Plan how customers will access your product</td>
<td>120</td>
</tr>
<tr>
<td>SaaS product guidelines</td>
<td>121</td>
</tr>
<tr>
<td>Product setup guidelines</td>
<td>121</td>
</tr>
<tr>
<td>Customer information requirements</td>
<td>121</td>
</tr>
<tr>
<td>Product usage guidelines</td>
<td>122</td>
</tr>
<tr>
<td>Architecture guidelines</td>
<td>122</td>
</tr>
<tr>
<td>Pricing SaaS products</td>
<td>122</td>
</tr>
<tr>
<td>Pricing for SaaS subscriptions</td>
<td>123</td>
</tr>
<tr>
<td>Pricing for SaaS contracts</td>
<td>124</td>
</tr>
<tr>
<td>SaaS customer onboarding</td>
<td>126</td>
</tr>
<tr>
<td>Configuring your SaaS product to accept new buyers</td>
<td>127</td>
</tr>
<tr>
<td>Amazon SNS notifications for SaaS products</td>
<td>128</td>
</tr>
<tr>
<td>SaaS product Amazon SNS message body</td>
<td>129</td>
</tr>
<tr>
<td>SaaS product Amazon SNS actions</td>
<td>129</td>
</tr>
<tr>
<td>Subscribing an SQS queue to the SNS topic</td>
<td>129</td>
</tr>
<tr>
<td>Polling the SQS queue for notifications</td>
<td>129</td>
</tr>
<tr>
<td>Accessing the AWS Marketplace Metering and Entitlement Service APIs</td>
<td>129</td>
</tr>
<tr>
<td>Metering for usage</td>
<td>130</td>
</tr>
<tr>
<td>Checking entitlements</td>
<td>131</td>
</tr>
<tr>
<td>SaaS product integration checklist</td>
<td>132</td>
</tr>
<tr>
<td>Reporting</td>
<td>134</td>
</tr>
<tr>
<td>Code examples</td>
<td>134</td>
</tr>
<tr>
<td>ResolveCustomer code example</td>
<td>135</td>
</tr>
<tr>
<td>GetEntitlement code example</td>
<td>135</td>
</tr>
<tr>
<td>BatchMeterUsage code example</td>
<td>136</td>
</tr>
<tr>
<td>Using AWS PrivateLink with AWS Marketplace</td>
<td>137</td>
</tr>
<tr>
<td>Introduction</td>
<td>137</td>
</tr>
<tr>
<td>Configuring your product</td>
<td>138</td>
</tr>
<tr>
<td>Submitting your product</td>
<td>139</td>
</tr>
<tr>
<td>Buyer access to VPC endpoints</td>
<td>139</td>
</tr>
<tr>
<td>Appendix: Checklists</td>
<td>140</td>
</tr>
<tr>
<td>Data products</td>
<td>141</td>
</tr>
<tr>
<td>Submitting your product</td>
<td>142</td>
</tr>
<tr>
<td>Using the Products tab</td>
<td>143</td>
</tr>
<tr>
<td>Company and product logo requirements</td>
<td>143</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Requirements for submitting paid repackaged software</td>
<td>144</td>
</tr>
<tr>
<td>AMI cloning and product code assignment</td>
<td>147</td>
</tr>
<tr>
<td>Submitting your product</td>
<td>145</td>
</tr>
<tr>
<td>AMI self-service scanning</td>
<td>146</td>
</tr>
<tr>
<td>AMI cloning and product code assignment</td>
<td>147</td>
</tr>
<tr>
<td>IAM policy for AMI products</td>
<td>211</td>
</tr>
<tr>
<td>IAM policy for SaaS products</td>
<td>211</td>
</tr>
<tr>
<td>IAM for AWS Marketplace</td>
<td>203</td>
</tr>
<tr>
<td>Creating users</td>
<td>204</td>
</tr>
<tr>
<td>Creating or using groups</td>
<td>205</td>
</tr>
<tr>
<td>Signing in as an IAM user</td>
<td>206</td>
</tr>
<tr>
<td>Policies and permissions for AWS Marketplace sellers</td>
<td>207</td>
</tr>
<tr>
<td>Policies</td>
<td>207</td>
</tr>
<tr>
<td>Permissions</td>
<td>207</td>
</tr>
<tr>
<td>AWS Marketplace Commerce Analytics Service account permissions</td>
<td>209</td>
</tr>
<tr>
<td>AWS Marketplace Product Support Connection account permissions</td>
<td>209</td>
</tr>
<tr>
<td>Amazon SQS permissions</td>
<td>210</td>
</tr>
<tr>
<td>AWS Marketplace metering and entitlement API permissions</td>
<td>211</td>
</tr>
<tr>
<td>IAM policy for SaaS products</td>
<td>211</td>
</tr>
<tr>
<td>IAM policy for AMI products</td>
<td>211</td>
</tr>
<tr>
<td>AWS Marketplace Seller Guide</td>
<td></td>
</tr>
<tr>
<td>Seller reports and data feeds</td>
<td>153</td>
</tr>
<tr>
<td>Accessing reports</td>
<td>153</td>
</tr>
<tr>
<td>Daily business report</td>
<td>154</td>
</tr>
<tr>
<td>Daily customer subscriber report</td>
<td>161</td>
</tr>
<tr>
<td>Disbursement report</td>
<td>163</td>
</tr>
<tr>
<td>Monthly billed revenue report</td>
<td>168</td>
</tr>
<tr>
<td>Sales compensation report</td>
<td>174</td>
</tr>
<tr>
<td>US sales and use tax report</td>
<td>176</td>
</tr>
<tr>
<td>Daily ref tag</td>
<td>179</td>
</tr>
<tr>
<td>Weekly ref tag</td>
<td>180</td>
</tr>
<tr>
<td>Data feeds</td>
<td>181</td>
</tr>
<tr>
<td>Storage and structure of data feeds</td>
<td>181</td>
</tr>
<tr>
<td>Accessing data feeds</td>
<td>182</td>
</tr>
<tr>
<td>Using data feeds</td>
<td>183</td>
</tr>
<tr>
<td>Account data feed</td>
<td>183</td>
</tr>
<tr>
<td>Address data feed</td>
<td>184</td>
</tr>
<tr>
<td>Billing event data feed</td>
<td>186</td>
</tr>
<tr>
<td>Legacy mapping data feed</td>
<td>194</td>
</tr>
<tr>
<td>Offer data feed</td>
<td>194</td>
</tr>
<tr>
<td>Offer product data feed</td>
<td>195</td>
</tr>
<tr>
<td>Offer target data feed</td>
<td>196</td>
</tr>
<tr>
<td>Product data feed</td>
<td>197</td>
</tr>
<tr>
<td>Tax item data feed</td>
<td>198</td>
</tr>
<tr>
<td>Security</td>
<td>203</td>
</tr>
<tr>
<td>IAM for AWS Marketplace</td>
<td>203</td>
</tr>
<tr>
<td>Creating users</td>
<td>204</td>
</tr>
<tr>
<td>Creating or using groups</td>
<td>205</td>
</tr>
<tr>
<td>Signing in as an IAM user</td>
<td>206</td>
</tr>
<tr>
<td>Policies and permissions for AWS Marketplace sellers</td>
<td>207</td>
</tr>
<tr>
<td>Policies</td>
<td>207</td>
</tr>
<tr>
<td>Permissions</td>
<td>207</td>
</tr>
<tr>
<td>AWS Marketplace Commerce Analytics Service account permissions</td>
<td>209</td>
</tr>
<tr>
<td>AWS Marketplace Product Support Connection account permissions</td>
<td>209</td>
</tr>
<tr>
<td>Amazon SQS permissions</td>
<td>210</td>
</tr>
<tr>
<td>AWS Marketplace metering and entitlement API permissions</td>
<td>211</td>
</tr>
<tr>
<td>IAM policy for SaaS products</td>
<td>211</td>
</tr>
<tr>
<td>IAM policy for AMI products</td>
<td>211</td>
</tr>
</tbody>
</table>
What Is AWS Marketplace?

AWS Marketplace is a curated digital catalog customers can use to find, buy, deploy, and manage third-party software, data, and services that customers need to build solutions and run their businesses. AWS Marketplace includes thousands of software listings from popular categories such as security, networking, storage, machine learning, business intelligence, database, and DevOps. AWS Marketplace also simplifies software licensing and procurement with flexible pricing options and multiple deployment methods. In addition, AWS Marketplace includes data products available from AWS Data Exchange.

Customers can quickly launch preconfigured software with just a few clicks, and choose software solutions in Amazon Machine Images (AMIs), software as a service (SaaS), and other formats. You can browse and subscribe to data products. Flexible pricing options include free trial, hourly, monthly, annual, multi-year, and BYOL, and get billed from one source. AWS handles billing and payments, and charges appear on customers’ AWS bill.

You can use AWS Marketplace as a buyer (subscriber), seller (provider), or both. Anyone with an AWS account can use AWS Marketplace as a buyer, and can register to become a seller. A seller can be an independent software vendor (ISV), value-added reseller, or individual who has something to offer that works with AWS products and services.

**Note**

Data product providers must meet the AWS Data Exchange eligibility requirements. For more information, see Providing Data Products on AWS Data Exchange in the AWS Data Exchange User Guide.

Every software product on AWS Marketplace has been through a curation process. On the product page, there can be one or more offerings for the product. When the seller submits a product in AWS Marketplace, they define the price of the product and the terms and conditions of use. When a consumer subscribes to a product offering, they agree to the pricing and terms and conditions set for the offer.

The product can be free to use or it can have an associated charge. The charge becomes part of your AWS bill, and after you pay, AWS Marketplace pays the seller. Products can take many forms. For example, a product can be offered as an Amazon Machine Image (AMI) that is instantiated using your AWS account. The product can also be configured to use AWS CloudFormation templates for delivery to the consumer. The product can also be software as a service (SaaS) offerings from an ISV, web ACL, set of rules, or conditions for AWS WAF.

Software products can be purchased at the listed price using the ISV’s standard end user license agreement (EULA) or offered with customer pricing and EULA. Products can also be purchased under a contract with specified time or usage boundaries. After the product subscriptions are in place, the consumer can copy the product to their AWS Service Catalog to manage how the product is accessed and used in the consumer’s organization.

Using AWS Marketplace as a seller

As a seller, go to the AWS Marketplace Management Portal to register. If you’re charging for use of your product, you must also provide tax and banking information as part of your registration. When you register, you create a profile for your company or for yourself that is discoverable on AWS Marketplace. You also use the AWS Marketplace Management Portal to create and manage product pages for your products.

Eligible partners can programmatically list AWS Marketplace products outside of AWS Marketplace. For information about becoming an eligible partner, contact your AWS Marketplace business development partner.
For more information about creating the product types in AWS Marketplace, see the following:

- AWS Marketplace for Desktop Applications (AMDA) (p. 58)
- AMI-based products (p. 61)
- Container-based products (p. 85)
- Machine learning products (p. 106)
- Software as a service (SaaS)-based products (p. 113)

For more information about data products, see What Is AWS Data Exchange? in the AWS Data Exchange User Guide.

After you are registered as a seller, you use the AWS Marketplace Management Portal to manage your account and products, and to access usage reports for your products.
Getting started as a seller

If you want to sell your software in AWS Marketplace, review the requirements and then follow the
steps to register as a seller. There are different registration requirements based on where you reside and
what type of products you're selling. To register as a seller in AWS Marketplace, you can use an existing
AWS account or create a new account. All AWS Marketplace interactions are tied to the account that you
choose.

Notes

• Registering as an AWS Marketplace seller is a prerequisite to be an AWS Data Exchange data
provider. For more information about these requirements, see Providing Data Products on
AWS Data Exchange in the AWS Data Exchange User Guide.
• For information about the permissions that AWS Marketplace sellers need, see Policies and
permissions for AWS Marketplace sellers (p. 207).

Seller requirements for publishing free software
products

Whether you charge for your product or not, when you offer it in AWS Marketplace, you're selling that
product. The cost to the customer is $0.00, but you and the customer agree to a mutual contract for use
of the product. If you offer only free products, you don't have to provide banking information to AWS
Marketplace. To create and offer free products in AWS Marketplace, you must:

• Sell publicly available, full-feature production-ready software.
• Have a defined customer support process and support organization.
• Provide a means to keep software regularly updated and free of vulnerabilities.
• Follow best practices and guidelines when marketing your product in AWS Marketplace.
• Be an AWS customer in good standing and meet the requirements in the terms and conditions for AWS
Marketplace sellers.

Additional seller requirements for paid products

If you charge for your products or offer Bring-Your-Own-License model (BYOL) products, you must also
meet the following requirements and provide this additional information:

• You must be a permanent resident or citizen in an eligible jurisdiction (p. 4), or a business entity
organized or incorporated in one of those areas.
• You must provide tax and bank account information. For US-based entities, a W-9 form and a banking
account from a US-based bank are required.
• Non-US sellers are required to provide a (i) W-8 form, value-added tax (VAT) or goods and services tax
(GST) registration number, and (ii) US bank information. If you don't have a US bank account, you can
register for a virtual US bank account from Hyperwallet.
• To provide data products, you must also request on-boarding through the Create case wizard for AWS
Support.
Eligible jurisdictions for paid products

To sell paid software in AWS Marketplace, you must be a permanent resident or citizen in one of the following countries, or a business entity organized or incorporated therein:

- Australia¹
- Bahrain¹ ²
- European Union (EU) member state¹
- Japan² ³
- New Zealand¹
- Norway¹ ²
- Switzerland¹ ²
- United Arab Emirates (UAE)¹ ²
- United Kingdom (UK)¹
- United States (US)

¹ Sellers of paid products in these countries must provide VAT registration information in country of establishment.
² If the buyer is in these countries, sellers may be responsible for tax invoicing and collections. Please consult with your tax advisor.
³ Sellers based in Japan have an obligation to self-account for the Japanese Consumption Tax (JCT) on the listing fee charges.

For more information about VAT, invoicing, and your tax obligations as a seller, see AWS Marketplace Sellers on Amazon Web Service Tax Help.

AWS Marketplace Management Portal

The AWS Marketplace Management Portal is the tool that you use to register as an AWS Marketplace seller and then to manage the products that you sell in AWS Marketplace. You can complete the following tasks on the portal:

- Register as an AWS Marketplace seller.
- Use the Products page to submit new software products and update existing software products.
- Monitor the status of your requests.
- Upload files needed to create and manage your new software products.
- Manage your software products into incremental channel revenue by taking advantage of the go-to-market activities.
- Measure the results of your marketing efforts within hours of launch, including the usage and revenue driven by your campaigns.
- Enable customer service representatives to retrieve customer data in real time.
- Initiate an automatic AMI scan to detect vulnerabilities.
Note

Data products are published and managed from the AWS Data Exchange console. AWS Data Exchange providers can use the AWS Marketplace Management Portal to register as a seller, request AWS Data Exchange on-boarding, access seller reports, and submit refund requests.

All registered sellers can access the AWS Marketplace Management Portal using their AWS credentials for the account that they used to create their products. The account that you use is defined as the seller of record when a customer subscribes to your product. If you need help determining the specific account that is the seller of record for your products, contact the AWS Marketplace Seller Operations team.

AWS Marketplace strongly recommends using AWS Identity and Access Management (IAM) roles to sign in to the AWS Marketplace Management Portal rather than using your root account credentials. For more information, see IAM Users in the IAM User Guide.

To enable people in your company to sign in to the AWS Marketplace Management Portal, create an IAM user for each person you want to have access and define access permissions to the AWS Marketplace Management Portal. We also recommend creating a root or account owner IAM to use for access.

Seller registration process

To register as an AWS Marketplace seller, from the AWS Marketplace Management Portal (AMMP), choose **Sign Up as an AWS Marketplace Seller** and accept the terms and conditions. Identify an AWS account to use as your primary AWS Marketplace account. You can use an existing account or register a new AWS account so long as the account is linked to a valid payment method. This account will be the seller of record for your products in AWS Marketplace and will be used for reporting, disbursement, and communication from AWS Marketplace to you.

Note

Once you use an AWS account to list a product on AWS Marketplace, you cannot change the AWS account associated with the product.

You can change other product information (name, website, description) in AWS Marketplace after the product is created. You can also use AWS Identity and Access Management (AWS IAM) to configure your primary AWS account to allow multiple users with various permissions to access the AWS Marketplace Management Portal. For more information, visit the section called “IAM for AWS Marketplace” (p. 203).

US bank account for sellers of paid products

A US bank account is required for all sellers who want to sell paid software in AWS Marketplace. AWS Marketplace only disburses to US bank accounts.

Note

For a list of countries where you can offer paid products in AWS Marketplace, see Eligible jurisdictions for paid products (p. 4).

If you do not already have a US bank account, you may be able to obtain one through Hyperwallet. Hyperwallet can provide you with a US account, which you can provide to AWS Marketplace for your AWS Marketplace disbursements.

Hyperwallet is an independent service provider that may enable you to transfer funds to another bank account in a supported currency. For a limited time, you will not be required to pay certain Hyperwallet service fees in connection with AWS Marketplace disbursements.

- By adding your Hyperwallet account details to your AWS Marketplace seller account, you agree and acknowledge that AWS Marketplace will share your name, email address, and account number with Hyperwallet to confirm your status as an AWS Marketplace seller.
Additional fees may apply to your use of Hyperwallet services (including transfer fees and foreign exchange fees required to transfer funds into your local currency), as well as foreign exchange rates. Hyperwallet’s service fee will be waived for a limited time, and only with respect to AWS Marketplace disbursements of the proceeds from your Paid products into your Hyperwallet account. Consult the Fees section of the Hyperwallet site or contact Hyperwallet for more information and to review applicable fees.

To begin registration with Hyperwallet and obtain your US bank account information

1. Use the URL and PIN emailed to you by AWS Marketplace to register with Hyperwallet. You will receive the email as part of your registration process.
2. After you have activated your Hyperwallet account, follow the steps described on the Hyperwallet registration portal to complete registration and receive your deposit account information.
3. When you have obtained an account from Hyperwallet, add your Hyperwallet account information to your AWS account using the Bank Account Registration Tool.

AWS Marketplace Tax Calculation Service

AWS Marketplace Tax Calculation Service provides the ability to calculate and collect US sales and use tax for existing and new products. Some states are not eligible for Tax Calculation Service because AWS Marketplace is required by law to collect and remit applicable sales tax attributable to taxable sales of your products to customers based in these states. To use the service, configure your tax nexus settings for your seller profile, and then assign product tax codes to your products.

To configure your tax nexus settings, open the AWS Marketplace Management Portal, and under the Settings tab configure the applicable tax nexus settings. Then, assign product tax codes (PTCs) to your products through the AWS Marketplace Management Portal. We recommend that you review the AWS Marketplace Tax Methodology and AWS Marketplace Product Tax Code Guidance in their entirety before completing this process. For product types not supported by the Products tab submission process, download a product load form by choosing File Upload from the Assets tab. You must edit and upload the updated product load form.

Once you have completed these two steps, US sales and use tax calculation will be enabled. Note the following:

- Activation of your tax nexus settings takes from five to 48 hours.
- Tax nexus settings must be configured before you can assign PTCs.
- PTC assignment happens 24 hours after the AWS Marketplace team approves and publishes your product, which may take 3-5 days from the time you submit your product change request.
- When tax calculation begins, estimated sales tax charges will be included in customer invoices. Sales tax will be calculated based on factors including, but not limited to, the customer’s billing address, the tax code of your product, and your tax nexus settings. The resulting sales tax charge, if applicable, will be included in the customer’s invoice and identified as a US sales tax charge under the specific product sold by your company. Note that customer invoices will show your company’s Legal Name, which you provided when you registered to become an AWS Marketplace seller.
- The collected sales tax funds are sent with your monthly disbursement, and the US Sales and Use Tax Report is available to you on the fifteenth of the month, detailing what taxes were collected. You are responsible for remitting your own taxes.

If you enroll for the AWS Marketplace Tax Calculation Service, we also recommend that you register for the Amazon Tax Exemption Program (ATEP). You are not required to use this service. However, we recommend that all AWS Marketplace sellers who use the Tax Calculation Service participate in ATEP. Participation helps to reduce the number of tax-only refunds that will need to be processed to qualified customers registered in ATEP.
You can edit or delete the tax nexus information on the Tax Calculation Service Settings page in the AWS Marketplace Management Portal.

For more information, visit AWS Marketplace Sellers on Amazon Web Service Tax Help to learn more about where AWS collects sales tax, VAT, or GST on your sales and remits such taxes to the local tax authorities, in the name of AWS, Inc.

Disbursement

- A valid payment method, a registered US bank account, and a submitted W9 form are required for disbursement.
- Sellers of paid products are required to provide a W-8, value added tax (VAT) or good and services tax (GST) registration number, and a US bank account. Hyperwallet can provide you with a US bank account, which you can provide to AWS Marketplace for your AWS Marketplace disbursements.
- AWS disburses payments monthly directly to the bank account associated with the seller account, minus AWS Marketplace service fees. Payment is disbursed on a rolling monthly basis based off of when the seller account was created, not the beginning of each month.
- Funds are disbursed only after they are collected from the customer.
- If you participate in the AWS Marketplace Tax Calculation Service, any US sales and use tax collected from customers will be included in your monthly disbursement.

Already a seller?

Manage your products into incremental channel revenue by taking advantage of the go-to-market activities made available in the AWS Marketplace Management Portal. Activities include the following:

- Measure the results of your marketing efforts within hours, including the usage and revenue driven by your campaigns.
- Enable customer service representatives to retrieve customer data in real time.
- Upload files needed to create and manage your products, and monitor progress as we process them.

Seller toolkit

The AWS Marketplace Management Portal is your primary tool for selling products on AWS Marketplace. The following additional tools can give you more insight into your customer base and help you better understand your sales.

- AWS Marketplace Commerce Analytics Service (p. 7)
- AWS Marketplace Enhanced Data Sharing Program (p. 18)
- AWS Marketplace Field Demonstration Program (p. 19)
- Product Support Connection (p. 19)
- Seller reports and data feeds (p. 153)

AWS Marketplace Commerce Analytics Service

The AWS Marketplace Commerce Analytics Service lets you programmatically access product and customer data through AWS Marketplace. After you enroll in the service, you can access your usage, subscription, and billing reports through the AWS SDK.
The data you request using the SDK tools is delivered to your AWS account as datasets. Most of the datasets correspond to the same data as the text-based reports available on the AWS Marketplace Management Portal. You can request datasets for a specific date, and the data is delivered to the provided Amazon S3 bucket. Notification of data delivery is provided by the Amazon Simple Notification Service (Amazon SNS).

Terms and conditions

These AWS Marketplace Commerce Analytics Service Terms and Conditions (these "CAS Terms") contain the terms and conditions specific to your use of and access to the AWS Marketplace Commerce Analytics Service ("CA Service") and are effective as of the date you click an "I Accept" button or check box presented with these CAS Terms or, if earlier, when you use any CA Service offerings. These CAS Terms are an addendum to the Terms and Conditions for AWS Marketplace Sellers (the "AWS Marketplace Seller Terms") between you and Amazon Web Services, Inc. ("AWS," "we," "us" or "our"), the terms of which are hereby incorporated herein. In the event of a conflict between these CAS Terms and the AWS Marketplace Seller Terms, the terms and conditions of these CAS Terms apply, but only to the extent of such conflict and solely with respect to your use of the CA Service. Capitalized terms used herein but not defined herein shall have the meanings set forth in the AWS Marketplace Seller Terms.

1. **CA Services and CAS Data.** To qualify for access to the CA Service, you must be an AWS Marketplace Seller bound by existing AWS Marketplace Seller Terms. Information and data you receive or have access to in connection with the CA Service ("CAS Data") constitutes Subscriber Information and is subject to the restrictions and obligations set forth in the AWS Marketplace Seller Terms. You may use CAS Data on a confidential basis to improve and target marketing and other promotional activities related to Your AWS Marketplace Content provided that you do not (a) disclose CAS Data to any third party; (b) use any CAS Data in any way inconsistent with applicable privacy policies or law; (c) contact a subscriber to influence them to make an alternative purchase outside of the AWS Marketplace; (d) disparage us, our affiliates, or any of their or our respective products; or (e) target communications of any kind on the basis of the intended recipient being an AWS Marketplace subscriber.

2. **CA Service Limitations and Security.** You will only access (or attempt to access) the CA Service by the means described in the CA Service documentation. You will not misrepresent or mask your identity or your client’s identity when using the CA Service. We reserve the right, in our sole discretion, to set and enforce limits on your use of the CA Service, including, without limitation, with respect to the number of connections, calls and servers permitted to access the CA Service during any period of time. You agree to, and will not attempt to circumvent such limitations. We reserve the right to restrict, suspend or terminate your right to access the CA Service if we believe that you may be in breach of these CAS Terms or are misusing the CA Service.
3. **CA Service Credential Confidentiality and Security.** CA Service credentials (such as passwords, keys and client IDs) are intended to be used by you to identify your API client. You are solely responsible for keeping your credentials confidential and will take all reasonable measures to avoid disclosure, dissemination or unauthorized use of such credentials, including, at a minimum, those measures you take to protect your own confidential information of a similar nature. CA Service credentials may not be embedded on open source projects. You are solely responsible for any and all access to the CA Service with your credentials.

4. **Modification.** We may modify these CAS Terms at any time by posting a revised version on the AWS Site or providing you with notice in accordance with the AWS Marketplace Seller Terms. The modified terms will become effective upon posting or, if we notify you by email, as stated in the email message. By continuing use or access the CA Service after the effective date of any modifications to these CAS Terms, you agree to be bound by the modified terms.

5. **Termination.** These CAS Terms and the rights to use CAS Data granted herein will terminate, with or without notice to you upon termination of your AWS Marketplace Seller Terms for any reason. In addition, we may stop providing the CA Services or terminate your access to the CA Services at any time for any or no reason.

### Onboarding guide

You must configure your AWS account and AWS services to use the AWS Marketplace Commerce Analytics Service.

#### To use the AWS Marketplace Commerce Analytics Service

1. **Set up your AWS account with permissions** (p. 9).
2. **Create a destination Amazon S3 bucket** (p. 9).
3. **Configure an Amazon SNS topic for response notifications** (p. 9).
4. **Enroll in the Commerce Analytics Service program** (p. 10).
5. **Verify your configuration** (p. 10).

#### Set up your AWS account with permissions

AWS Marketplace **strongly** recommends using AWS Identity and Access Management (IAM) roles to sign in to the AWS Marketplace Management Portal rather than using your root account credentials. See the section called “Policies and permissions for AWS Marketplace sellers” (p. 207) for specific IAM permissions for AWS Marketplace Commerce Analytics Service permissions. See Create IAM Users for details. By creating individual IAM users for people accessing your account, you can give each IAM user a unique set of security credentials. You can also grant different permissions to each IAM user. If necessary, you can change or revoke an IAM user’s permissions any time.

#### Create a destination Amazon S3 bucket

The Commerce Analytics Service delivers the data you request to an Amazon S3 bucket that you specify. If you already have an Amazon S3 bucket to use, proceed to the next step.

If you don’t have an Amazon S3 bucket or you want to create an Amazon S3 bucket specifically for this data, see How do I Create an S3 Bucket.

#### Configure an Amazon SNS topic for response notifications

The Commerce Analytics Service delivers response notifications using Amazon SNS. The service publishes messages to this topic to notify you when your datasets are available or if an error occurred. If you already have an Amazon SNS topic for this purpose, proceed to the next step.
If you don't have an Amazon SNS topic configured for this service, configure one now. For instructions, see Create a Topic.

Record the topic Amazon Resource Name (ARN) for the topic you created, because the ARN is required to call the service.

Enroll in the Commerce Analytics Service program

The Commerce Analytics Service accesses the Amazon S3 bucket and Amazon SNS topic after you configure the service with the ARN for the topic and name of the bucket.

To enable access

1. Log in to the AWS Marketplace Management Portal with the AWS account you use to manage your AWS Marketplace products.
2. Navigate to the Commerce Analytics Service enrollment page.
3. Enter the Amazon S3 bucket name and Amazon SNS topic ARN, and choose Enroll.
4. On the permissions page, choose Allow.
5. On the AWS Marketplace Management Portal, record the Role Name ARN in the success message. You need the ARN to call the service.

Verify your configuration

The last step is to verify that your configuration works as expected.

To test your configuration

1. Download, install, and configure the AWS Command Line Interface (AWS CLI).
2. Using the AWS CLI, run this command.
   
   ```bash
   aws marketplacecommerceanalytics generate-data-set \
   --data-set-type "customer_subscriber_hourly_monthly_subscriptions" \
   --data-set-publication-date "{TODAY'S-DATE}" \
   --role-name-arn "{YOUR-ROLE-NAME-ARN}" \
   --destination-s3-bucket-name "{YOUR-S3-BUCKET}" \
   --destination-s3-prefix "test-prefix" \
   --sns-topic-arn "{YOUR-SNS-TOPIC-ARN}"
   ```

   • For `--data-set-publication-date`, replace `{TODAY'S-DATE}` with the current date using ISO-8601 format, YYYY-MM-DDT00:00:00Z, where YYYY is the four-digit year, MM is the two-digit month, and DD is the two-digit day.
   • For `--role-name-arn`, replace `{YOUR-ROLE-NAME-ARN}` with the ARN of the role you received from the enrollment process in Enroll in the Commerce Analytics Service program (p. 10).
   • For `--destination-s3-bucket-name`, replace `{YOUR-S3-BUCKET}` with the Amazon S3 bucket you created in Create a destination Amazon S3 bucket (p. 9).
   • For `--sns-topic-arn`, replace `{YOUR-SNS-TOPIC-ARN}` with the Amazon SNS topic you created in Configure an Amazon SNS topic for response notifications (p. 9).

If you receive a response including the dataSetRequestId response from the service, you've completed the on-boarding process. A successful response looks like this:
Technical implementation guide

The AWS Marketplace Commerce Analytics Service is provided through the AWS SDK. This guide shows you how to interact with the service using the AWS CLI and the AWS SDK for Java.

IAM policy for Commerce Analytics Service

To allow your IAM users to use the Commerce Analytics Service, attach the following inline policy to your users.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "marketplacecommerceanalytics:GenerateDataSet",
      "Resource": "*"
    }
  ]
}
```

For more information, see Creating Policies in the IAM console in the IAM User Guide.

Making Requests with the AWS CLI

To get started, download the AWS CLI. The following AWS CLI example makes a request for the Hourly/Monthly Subscriptions dataset for October 1, 2017. This dataset is published to the demo-bucket Amazon S3 bucket using the prefix demo-prefix, and the notification message is delivered to the demo-topic Amazon SNS topic.

```
aws marketplacecommerceanalytics generate-data-set \
--data-set-type "customer_subscriber_hourly_monthly_subscriptions" \
--data-set-publication-date "2017-10-01T00:00:00Z" \
--role-name-arn "arn:aws:iam::123412341234:role/MarketplaceCommerceAnalyticsRole" \
--destination-s3-bucket-name "demo-bucket" \
--destination-s3-prefix "demo-prefix" \
```

This request returns an identifier that is unique for each request. You can use this identifier to correlate requests with notifications published to your Amazon SNS topic. The following example is an example of this identifier.

```
{
  "dataSetRequestId": "646dd4ed-6806-11e5-a6d8-fd5dbcaa74ab"
}
```

Making requests with the AWS SDK for Java

To start, download the AWS Java SDK. The following AWS SDK for Java example makes a request for the Hourly/Monthly Subscriptions dataset for October 1, 2015. This dataset is published to the demo-
bucket Amazon S3 bucket using the prefix demo-prefix, and the notification message is delivered to the demo-topic Amazon SNS topic.

```java
import java.text.SimpleDateFormat;
import java.util.Date;
import com.amazonaws.services.marketplacecommerceanalytics.AWSMarketplaceCommerceAnalyticsClient;
import com.amazonaws.services.marketplacecommerceanalytics.model.GenerateDataSetRequest;
import com.amazonaws.services.marketplacecommerceanalytics.model.GenerateDataSetResult;

/**
 * This sample demonstrates how to make basic requests to the AWS Marketplace Commerce
 * Analytics service using the AWS SDK for Java.
 */
public class MarketplaceCommerceAnalyticsSample {
    public static void main(String[] args) throws ParseException {
        
        AWSCredentials credentials = null;
        try {
            credentials = new ProfileCredentialsProvider().getCredentials();
        } catch (Exception e) {
            throw new AmazonClientException("Cannot load the credentials from the credential profiles 
            file. Make sure that your credentials file is at the correct 
            location (~/.aws/credentials), and is in valid format.", e);
        }
```
AWS Marketplace Seller Guide
AWS Marketplace Commerce Analytics Service

AWSMarketplaceCommerceAnalyticsClient client = new
AWSMarketplaceCommerceAnalyticsClient(credentials);
Region usEast1 = Region.getRegion(Regions.US_EAST_1);
client.setRegion(usEast1);
System.out.println("===================================================================");
System.out.println("Getting Started with AWS Marketplace Commerce Analytics Service");
System.out.println("==================================================================\n");
// Create a data set request with the desired parameters
GenerateDataSetRequest request = new GenerateDataSetRequest();
request.setDataSetType("customer_subscriber_hourly_monthly_subscriptions");
request.setDataSetPublicationDate(convertIso8601StringToDateUtc("2014-06-09T00:00:00Z"));
request.setRoleNameArn("arn:aws:iam::864545609859:role/MarketplaceCommerceAnalyticsRole");
request.setDestinationS3BucketName("awsmp-goldmine-seller");
request.setDestinationS3Prefix("java-sdk-test");
System.out.println("Creating a request for data set %s for publication date %s.",
request.getDataSetType(), request.getDataSetPublicationDate()));
try {
// Make the request to the service
GenerateDataSetResult result = client.generateDataSet(request);
// The Data Set Request ID is a unique identifier that you can use to correlate the
// request with responses on your Amazon SNS topic
System.out.println("Request successful, unique ID: " + result.getDataSetRequestId());
} catch (AmazonServiceException ase) {
System.out.println("Caught an AmazonServiceException, which means your request made it "
+ "to the AWS Marketplace Commerce Analytics service, but was rejected with an "
+ "error response for some reason.");
System.out.println("Error Message: " + ase.getMessage());
System.out.println("HTTP Status Code: " + ase.getStatusCode());
System.out.println("AWS Error Code: " + ase.getErrorCode());
System.out.println("Error Type: " + ase.getErrorType());
System.out.println("Request ID: " + ase.getRequestId());
} catch (AmazonClientException ace) {
System.out.println("Caught an AmazonClientException, which means the client encountered "
+ "a serious internal problem while trying to communicate with the AWS Marketplace"
+ "Commerce Analytics service, such as not being able to access the "
+ "network.");
System.out.println("Error Message: " + ace.getMessage());
}
private static Date convertIso8601StringToDateUtc(String dateIso8601) throws ParseException
{
TimeZone utcTimeZone = TimeZone.getTimeZone("UTC");
DateFormat utcDateFormat = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ssX");
utcDateFormat.setTimeZone(utcTimeZone);
return utcDateFormat.parse(dateIso8601);
}
You should expect results similar to this example.

==================================================================
Getting Started with AWS Marketplace Commerce Analytics Service
==================================================================

Creating a request for data set customer_subscriber_hourly_monthly_subscriptions for
publication
date Sun Jun 08 17:00:00 PDT 2014.
Request successful, unique ID: c59aff81-6875-11e5-a6d8-fd5dbcaa74ab
Technical documentation

The service exposes one method, `GenerateDataSet`, which enables you to request datasets to be published to your Amazon S3 bucket. The following table lists the parameters for `GenerateDataSet`.

### Dataset parameters

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set Type</td>
<td>This dataset will be returned as the result of the request.</td>
</tr>
<tr>
<td>Data Set Publication Date</td>
<td>The date a dataset was published.</td>
</tr>
<tr>
<td></td>
<td>For daily datasets, provide a date with day-level granularity for the desired day.</td>
</tr>
<tr>
<td></td>
<td>For monthly datasets, provide a date with month-level granularity for the desired month. The day value is ignored.</td>
</tr>
<tr>
<td>Role Name ARN</td>
<td>The ARN of the role with an attached permissions policy that provides the service with access to your resources.</td>
</tr>
<tr>
<td>Destination Amazon S3 Bucket Name</td>
<td>The name (the friendly name, not the ARN) of the destination Amazon S3 bucket. Your datasets are published to this location.</td>
</tr>
<tr>
<td>Destination Amazon S3 Prefix</td>
<td>(Optional) The Amazon S3 prefix for the published dataset, similar to a directory path in standard file systems.</td>
</tr>
<tr>
<td></td>
<td>For example, if given the bucket name <code>mybucket</code> and the prefix <code>myprefix/mydatasets</code>, the output file is published to <code>s3://DOC-EXAMPLE-BUCKET/myprefix/mydatasets/outputfile</code>.</td>
</tr>
<tr>
<td></td>
<td>If the prefix directory structure doesn't exist, it's created.</td>
</tr>
<tr>
<td></td>
<td>If no prefix is provided, the dataset is published to the Amazon S3 bucket root.</td>
</tr>
<tr>
<td>SNS Topic ARN</td>
<td>The ARN for the Amazon SNS topic that is notified when the dataset has been published or if an error occurs.</td>
</tr>
</tbody>
</table>

### Responses

The AWS Marketplace Commerce Analytics service returns two responses. The first is synchronous, which is returned immediately, and the second is asynchronous, which is returned using the Amazon SNS. The synchronous response is similar to this example.
**Data set parameters**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set Request ID</td>
<td>A unique identifier representing a specific request to the service. This identifier can be used to correlate a request with notifications on the Amazon SNS topic.</td>
</tr>
</tbody>
</table>

The asynchronous response is posted as a JSON-formatted document to your Amazon SNS topic and is similar to this example.

**Dataset parameters**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set S3 Location</td>
<td>The bucket name and key for the delivered dataset.</td>
</tr>
<tr>
<td>Data Set Meta Data S3 Location</td>
<td>The bucket name and key for the delivered dataset metadata file.</td>
</tr>
<tr>
<td>Data Set Request ID</td>
<td>A unique identifier representing a specific request to the service. This identifier can be used to correlate a request with notifications on the Amazon SNS topic.</td>
</tr>
<tr>
<td>Success</td>
<td>&quot;True&quot; if the operation succeeded; &quot;false&quot; if not.</td>
</tr>
<tr>
<td>Message</td>
<td>(Optional) If an error occurred (for example, &quot;Success&quot; is &quot;false&quot;), this message contains information about the failure.</td>
</tr>
</tbody>
</table>

**Example JSON-formatted asynchronous response**

```json
{
  "dataSetS3Location":{
    "bucketName": "demo-bucket",
    "key": "demo-prefix/customer_subscriber_hourly_monthly_subscriptions_2014-06-09.csv"
  },
  "dataSetMetaDataS3Location":{
    "bucketName": "demo-bucket",
    "key": "demo-prefix/customer_subscriber_hourly_monthly_subscriptions_2014-06-09.meta.json"
  },
  "dataSetRequestId": "f65b7244-6862-11e5-80e2-c5127e17c023",
  "success": true
}
```

**Outputs**

After a successful request, the requested dataset is delivered to your Amazon S3 bucket as a .csv file. A JSON-formatted metadata file is published to the same location as the dataset file. The metadata file provides useful information about the dataset and original request parameters. The metadata file has the same name as the dataset file, but ends with the extension .meta.json. The following table lists the metadata fields in the .csv file.
Metadata fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set Request ID</td>
<td>A unique identifier representing a specific request to the service. This identifier can be used to correlate a request with notifications on the Amazon SNS topic.</td>
</tr>
<tr>
<td>Data Set Coverage Range</td>
<td>Defines the start date/time and end date/time for the data coverage range. These dates are in ISO 8601 format.</td>
</tr>
<tr>
<td>Data Set Request Parameters</td>
<td>The original request parameters to the GenerateDataSet method.</td>
</tr>
<tr>
<td>Data Set S3 Location</td>
<td>The bucket name and key for the delivered dataset.</td>
</tr>
<tr>
<td>Data Set Meta Data S3 Location</td>
<td>The bucket name and key for the delivered dataset metadata file.</td>
</tr>
</tbody>
</table>

Following is an example of JSON-formatted metadata contents.

```json
{
  "dataSetRequestId": "43d7137b-8a94-4042-a09d-c41e87f371c1",
  "dataSetCoverageRange": {
    "startDateTime": "2014-06-08T00:00:00.000Z",
    "endDateTime": "2014-06-08T23:59:59.000Z"
  },
  "dataSetRequestParameters": {
    "sellerAccountId": "123412341234",
    "dataType": "customer_subscriber_hourly_monthly_subscriptions",
    "dataSetPublicationDate": "2014-06-09T00:00:00.000Z",
    "roleNameArn": "arn:aws:iam::123412341234:role/MarketplaceCommerceAnalyticsRole",
    "destinationS3BucketName": "demo-bucket",
    "destinationS3Prefix": "demo_prefix/customer_subscriber_hourly_monthly_subscriptions",
  },
  "dataSetS3Location": {
    "bucketName": "demo-bucket",
    "key": "demo_prefix/customer_subscriber_hourly_monthly_subscriptions_2014-06-09.csv"
  },
  "dataSetMetaDataS3Location": {
    "bucketName": "demo-bucket",
    "key": "demo_prefix/customer_subscriber_hourly_monthly_subscriptions_2014-06-09.meta.json"
  }
}
```

For a complete list of available datasets, including availability dates, refer to the AWS SDK documentation.

### Troubleshooting

This sections describes solutions to issues you may encounter with using the AWS Marketplace Commerce Analytics Service.

**I can't access the service because of an allow list issue.**
If you’re not yet registered as a seller on the AWS Marketplace, visit AWS Marketplace Management Portal to register. If you have already registered as a seller on AWS Marketplace, contact the AWS Marketplace Seller Operations team.

I can’t request datasets for a date in the past, even though the SDK documentation says it should be available for this date.

Even though datasets are listed as being available for certain dates in the past, we have data only since the time that you joined AWS Marketplace. If you believe that this is in error, contact the AWS Marketplace Seller Operations team.

When I call the service, I receive the error message “Could not connect to the endpoint URL: https://marketplacecommerceanalytics.eu-central-1.amazonaws.com/”

Currently, the AWS Marketplace Commerce Analytics Service is available only in the US East (N. Virginia) Region. You must make all calls to the Commerce Analytics Service to the us-east-1 endpoint.

If you’re using the AWS CLI, add the “--region” flag to each call and specify the AWS Region as us-east-1, as shown in the following example.

```
aws marketplacecommerceanalytics generate-data-set \
  --data-set-type "customer_subscriber_hourly_monthly_subscriptions" \
  --data-set-publication-date "2016-04-21T00:00:00Z" \
  --role-name-arn "arn:aws:iam::138136086619:role/MarketplaceCommerceAnalyticsRole" \
  --destination-s3-bucket-name "marketplace-analytics-service" \
  --destination-s3-prefix "test-prefix" \
  --sns-topic-arn "arn:aws:sns:eu-central-1:138136086619:Marketplace_Analytics_Service_Notice" \
  --region us-east-1
```

I want to use a different Amazon S3 bucket or Amazon SNS topic than the ones I selected when I went through the on-boarding process.

When enrolling in the AWS Marketplace Commerce Analytics Service, you specified an Amazon S3 bucket and Amazon SNS topic. The onboarding process configures your IAM permissions to allow the service access to only these specific resources. To use different resources, you need to modify your IAM policy:

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. Choose Roles on the left side of the IAM console.
3. Choose MarketplaceCommerceAnalyticsRole.
4. Expand the Inline Roles section, if not already expanded.
5. Locate the policy with a name that starts with oneClick_MarketplaceCommerceAnalyticsRole and choose Edit Policy.
6. In the policy document, locate the section that specifies actions related to the service that you want to modify. For example, to change your Amazon S3 bucket, locate the section that includes the actions that start with s3: and change their respective Resource selection to specify your new Amazon S3 bucket.

For additional information about IAM policies, see the following guide: https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html

I get an AccessDeniedException error when I call the GenerateDataSet action

This can happen if your IAM user doesn’t have the permissions necessary to call GenerateDataSet. The following procedure outlines the steps needed to update an IAM policy with those permissions using the IAM console.
To get the GenerateDataSet permissions

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/
2. From the navigation pane on the right, choose Users.
3. Choose the IAM user whose credentials you want to use for the marketplacecommerceanalytics AWS CLI commands to open the Summary page.
4. From the Permissions tab, choose Add inline policy
5. Open the JSON tab and paste the following code:

   ```json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
   "Action": "marketplacecommerceanalytics:GenerateDataSet",
   "Resource": "*",
   },
   ],
   }
   ```
6. Choose Review policy, provide the inline policy with a descriptive name, like GenerateDataSetPolicy, and choose Create policy.

After updating the permissions, run the AWS CLI command again with the same credentials as this IAM user to complete the action.

For more information, see Creating Policies in the IAM console in the IAM User Guide.

My problem isn't listed here.

Contact the AWS Marketplace Seller Operations team.

AWS Marketplace Enhanced Data Sharing Program

The focus of the AWS Marketplace Enhanced Data Sharing Program is to deliver buyer information, such as buyer email domain, AWS account ID, and location, on a daily and monthly basis to select AWS Marketplace sellers. The goal of delivering buyer information is to provide a framework for sellers to compensate their sales teams for AWS Marketplace subscription revenue. Sharing monthly billed revenue information, usage information, and disbursed funds information by buyer provides a mechanism to help you correctly map deals and opportunities to the appropriate sales representative by company, geography, and AWS account ID.

Requirements for participation

This program requires that the data is used only to activate and motivate AWS Marketplace seller field sales through formal sales compensation plans. For more information about the program, including how to enroll, contact us at <mpcustdesk@amazon.com>.

Enrollment requirements include:

- Annual pricing on all of your AWS Marketplace products.
- Sales compensation plans for all AWS Marketplace subscription revenue (hourly, monthly, annual, or metering).
- Formal announcement of an AWS Marketplace compensation plan.
- Agreement to use enhanced data sharing program data according to the terms outlined in Use of the data (p. 19).
Benefits for sellers

The benefits of enrolling in the enhanced data sharing program include:

- It increases the collective number of sales representatives driving awareness and adoption of AWS Marketplace subscriptions for your products.
- It provides incentive and compensation to your sales representatives working with AWS customers and prospects.
- It provides you with customer data to assist in analyzing, growing, and compensating sales teams for AWS Marketplace sell-through.
- It fosters a collaborative working relationship between the AWS sales and your sales team to better address customer needs.

Use of the data

The information shared with you as part of this program constitutes Amazon's Confidential Information under our nondisclosure agreement with you or, if no such agreement exists, the Terms and Conditions for AWS Marketplace Sellers. The purpose of our sharing this information is to allow you to evaluate the effectiveness of your marketing campaigns and communicate commission payments to your employees.

You may use such information for the foregoing purpose, including by sharing such information with employees who have a need to know such information to understand the source of commissions payable to them, provided that your use and sharing of such information complies with the confidentiality obligations in the agreements specified above, including, without limitation, Section 3.8 of the Terms and Conditions for AWS Marketplace Sellers.

AWS Marketplace Field Demonstration Program

The AWS Marketplace Field Demonstration Program (FDP) allows internally approved AWS employees to use some Independent Software Vendor (ISV) solutions through AWS Marketplace at no charge. Examples of approved AWS employees may include solutions architects, and sales and marketing professionals. The FDP allows these employees to demonstrate product capabilities for education and potential inclusion in customer workloads.

The FDP includes only products that ISVs contractually agreed to make available at no charge to the AWS field team for educational and demonstration purposes. The following product types are supported:

- Amazon Machine Images (AMIs) (p. 61)
- Containers (p. 85)
- Machine learning algorithms and model packages (SageMaker) (p. 106)
- Data sets (AWS Data Exchange)

As an ISV, you’re automatically enrolled in this program when you sign up as an AWS Marketplace seller. To opt out, submit a support request to the Managed Catalog Operations (MCO) team.

To view information about product usage under this program, see the AWS field demonstration usage (p. 173) section of the Monthly billed revenue report (p. 168).

Product Support Connection

AWS Marketplace Product Support Connection (PSC) is a feature that enables AWS Marketplace customers to provide contact information in the AWS Marketplace website for the purposes of obtaining and accessing product support from AWS Marketplace Sellers. AWS Marketplace shares the provided data
with participating Sellers via an API to enable a better support experience. Customers can choose to add contact details during or after a purchase of PSC-enabled AWS Marketplace products, and Sellers can retrieve the Customer contact data, along with relevant product subscription details, by calling a pull-based API.

Your staff can use the Customer Support Eligibility tool to access near-real-time information about a customer’s subscription to your products and provide fast, personalized service. AWS Marketplace Management Portal makes it easy to get started: Enter a customer’s AWS account ID to retrieve subscription and usage information from their account.

You also have the option to enroll your products in AWS Marketplace Product Support Connection (PSC). For products that are enrolled in PSC, AWS Marketplace customers can choose to provide contact information (including name, organization, email address, and phone number) via the AWS Marketplace website for the purposes of obtaining and accessing product support. If you enroll in PSC, AWS Marketplace shares the provided data with you via an API to help enable a more seamless support experience.

**Note**
Currently, data products don't support this feature.

### Technical implementation guide

This section covers API specification details and how to onboard with the product support connection feature. The PSC `start-support-data-export` API is part of the AWS Marketplace Commerce Analytics Service (CAS). To integrate with the API for PSC, you must first enroll in CAS. If you are already enrolled in CAS, use the same AWS Identity and Access Management (IAM) role that you created when you onboarded.

**IAM policy for PSC**

To allow your IAM users to access the AWS Marketplace product support connection feature, you must attach the following inline policy to your users.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "marketplacecommerceanalytics:StartSupportDataExport",
      "Resource": "*"
    },
  ]
}
```

For more information, see [Creating Policies in the IAM console](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_policies-inline.html) in the **IAM User Guide**.

### Making requests with the AWS Command Line Interface (CLI)

You can request an export of the PSC data using the AWS CLI or any of the AWS Software Development Kits (SDKs).

If you have already been using CAS to call the `generate-data-set` operation, you must use the same IAM role for both `generate-data-set` and `start-support-data-export`.

To ensure the security of the customer contact data available through the Product Support Connection program, we recommend that the Amazon Simple Storage Service (Amazon S3) bucket you use for `start-support-data-export` be separate from the S3 bucket you use for `generate-data-set`. Verify the permissions on your IAM role allow access to all S3 buckets you intend to use.
aws marketplacecommerceanalytics start-support-data-export
--data-set-type "test_customer_support_contacts_data" \ 
--from-date "{START-DATE}" \ 
--role-name-arn "{YOUR-ROLE-NAME-ARN}" \ 
--destination-s3-bucket-name "{YOUR-S3-BUCKET}" \ 
--destination-s3-prefix "test-prefix" \ 
--sns-topic-arn "{YOUR-SNS-TOPIC-ARN}"

A successful response from the service returns the dataSetRequestId of the request.

Example

```
{
  "dataSetRequestId": 
  "646dd4ed-6806-11e5-a6d8-fd5dbcaa74ab"
}
```

API request parameters and responses

StartSupportDataExport method

The StartSupportDataExport method allows you to request contact details that customers have submitted for your PSC-enabled products. Data is exported from the start date specified in the request up to 15 minutes prior to the time of the request. A successful request results in the dataset being published to the Amazon S3 bucket specified.

At this time, you can query the API to request the test_customer_support_contacts_data dataset. This will export a static test dataset containing data that does not correspond to any real customer data. You should use the test data for testing and integration. The customer_support_contacts_data option, which will return the real customer contact data for your PSC-enabled products, will not be available until after the General Availability of this feature later in 2016.

Request parameters

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set Type</td>
<td>The type of dataset requested to be exported. Valid options for datasets are:</td>
</tr>
<tr>
<td></td>
<td>test_customer_support_contacts_data</td>
</tr>
<tr>
<td></td>
<td>customer_support_contacts_data</td>
</tr>
</tbody>
</table>

The test_customer_support_contacts_data dataset provides sample data for testing and integration purposes and is available immediately. The customer_support_contacts_data dataset is currently unavailable. This option will contain actual customer data and be available upon general availability of PSC.

<p>| From Date | The earliest date of data to be exported. The exported data will contain information from the |</p>
<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specified From Date</td>
<td>From Date must be expressed as an ISO 8601 date/time string. If you would like to receive the full data set, as opposed to a set of updates, specify any date prior to the date when you onboarded to the program. To receive only incremental data since your last request, specify the endDateTime from the dataSetCoverageRange from the metadata JSON file resulting from your previous request. See below for more information about the metadata JSON file.</td>
</tr>
<tr>
<td>Role Name ARN</td>
<td>The Amazon Resource Name (ARN) of the IAM role with an attached permissions policy which provides the service with access to your resources.</td>
</tr>
<tr>
<td>Destination S3 Bucket Name</td>
<td>The name (friendly name, not ARN) of the destination Amazon S3 bucket. Your datasets will be published to this location.</td>
</tr>
<tr>
<td>Destination S3 Prefix</td>
<td>(Optional) The desired Amazon S3 prefix for the published dataset, similar to a directory path in standard file systems. For example, if given the bucket name &quot;mybucket&quot; and the prefix &quot;myprefix/mydatasets&quot;, the output file &quot;outputfile&quot; would be published to &quot;s3://DOC-EXAMPLE-BUCKET/myprefix/mydatasets/outputfile&quot;. If the prefix directory structure does not exist, it will be created. If no prefix is provided, the data set will be published to the Amazon S3 bucket root.</td>
</tr>
<tr>
<td>SNS Topic ARN</td>
<td>The Amazon Resource Name (ARN) for the Amazon SNS topic that will be notified when the data set has been published, or if an error occurs.</td>
</tr>
</tbody>
</table>

**Responses**

Calls to the API will immediately return a response with the Data Set Request ID.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set Request ID</td>
<td>A unique identifier representing a specific request to the service. This identifier can be used to correlate a request with notifications on the Amazon SNS topic.</td>
</tr>
</tbody>
</table>
An additional response containing metadata will be posted to the Amazon Simple Notification Service (Amazon SNS) topic specified in the original request. The contents of the post are detailed in the following table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set S3 Location</td>
<td>The bucket name and key for the delivered dataset.</td>
</tr>
<tr>
<td>Data Set Meta Data S3 Location</td>
<td>The bucket name and key for the delivered dataset meta file.</td>
</tr>
<tr>
<td>Data Set Request ID</td>
<td>A unique identifier representing a specific request to the service. This identifier can be used to correlate a request with notifications on the Amazon SNS topic.</td>
</tr>
<tr>
<td>Success</td>
<td>&quot;True&quot; if the operation succeeded; &quot;false &quot; if not.</td>
</tr>
<tr>
<td>Message</td>
<td>(Optional) If an error occurred (for example, &quot;Success&quot; is “false”), this message will contain information about the failure.</td>
</tr>
</tbody>
</table>

The metadata file is JSON-formatted and contains the following fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set Request ID</td>
<td>A unique identifier representing a specific request to the service. This identifier can be used to correlate a request with notifications on the Amazon SNS topic.</td>
</tr>
<tr>
<td>Data Set Coverage Range</td>
<td>Defines the start date / time and end date / time for the data coverage range. These dates are in ISO 8601 format.</td>
</tr>
<tr>
<td>Data Set Request Parameters</td>
<td>The original request parameters to the GenerateDataSet method.</td>
</tr>
<tr>
<td>Data Set S3 Location</td>
<td>The bucket name and key for the delivered dataset.</td>
</tr>
<tr>
<td>Data Set Meta Data S3 Location</td>
<td>The bucket name and key for the delivered dataset metadata file.</td>
</tr>
<tr>
<td>Request Received Date Time</td>
<td>The date/time at which the request was received, in ISO 8601 format.</td>
</tr>
<tr>
<td>Request Completed Date Time</td>
<td>The date/time at which the request was completed, in ISO 8601 format.</td>
</tr>
</tbody>
</table>

Example JSON-formatted metadata contents

```json
{
   "dataSetRequestId": "c3c84ee0-5aba-11e6-8d9c-235dc080841d",
   "dataSetCoverageRange": {
```
Output data format

The output data contains customer contact records, product code, product ID, subscription start date, and the AWS account ID of the customer. A summary of the fields is shown in the following table. Each output file contains a comma-separated header, followed by the records containing customer data and subscription information. Each record contains a "Create", "Update", or "Delete" operation type to indicate whether the record is newly created, modified, or deleted since the "From Date" indicated in the API request. The overall file format adheres to the RFC4180 standard.

If multiple operations have occurred on a record in the time frame specified by the "from-date" parameter API request, only the most recent data will be reflected or exported. For example, if a customer creates and then updates a record, the record returned will be different depending on the specified "from-date". If the "from-date" is prior to the date at which the record was created, only a CREATE record will be passed in the output data set, and the record will reflect the most recently entered details. If the "from-date" is after the record was created, but before it was updated, only an UPDATE record will be passed in the output data set. If the from-date is after the record was updated, no record will be passed. Likewise, if a customer creates and then deletes a record, only the "DELETE" will appear in the output file.

If you would like to receive the full dataset, as opposed to a set of updates, specify any date prior to the date when you onboarded to the program. To receive only incremental data since your last request, specify the endDateTime from the dataSetCoverageRange from the metadata JSON file resulting from your previous request.

<table>
<thead>
<tr>
<th>Field</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product ID</td>
<td>36-character hexadecimal string</td>
<td>Unique identifier for the product in AWS Marketplace (GUID). Required field; always appears in every record.</td>
</tr>
<tr>
<td>Product Code</td>
<td>25-character alphanumeric string</td>
<td>Unique identifier for the product, associated with billing and available in Amazon Elastic</td>
</tr>
<tr>
<td>Field</td>
<td>Format</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Compute Cloud (Amazon EC2) instance metadata.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Required field; always appears in every record.</td>
</tr>
<tr>
<td>Customer Guid</td>
<td>36-character hexadecimal string</td>
<td>Unique GUID identifying the customer contact data record. This will be unique for each record that appears in the output file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Required field; always appears in every record.</td>
</tr>
<tr>
<td>Subscription Guid</td>
<td>36-character hexadecimal string</td>
<td>Unique GUID corresponding to the customer's product subscription. A customer can have multiple subscriptions to the same product.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Required field; always appears in every record.</td>
</tr>
<tr>
<td>Subscription Start Date</td>
<td>ISO 8601 date/time, with UTC time zone. The format is YYYY-MM-DDTHH:mm:ss.nnnZ, where YYYY is year, MM is month, DD is day, HH is hour from 00-23, mm is minute of hour from 00-59, ss is second of minute from 00-59, and nnn is millisecond of second from 000-9999, such as “2016-04-07T14:05:15.275Z”</td>
<td>Start date of the customer's product subscription.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Required field; always appears in every record.</td>
</tr>
<tr>
<td>Organization</td>
<td>String with a maximum length of 255 characters</td>
<td>Organization name provided by the customer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Always appears in records with operation type &quot;Update&quot; or &quot;Create.&quot; Does not appear in records with operation type &quot;Delete.&quot;</td>
</tr>
<tr>
<td>AWS Customer Id</td>
<td>12-digit numeric string which may include leading zeroes</td>
<td>The AWS customer ID for the customer subscribed to the product.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Required field; always appears in every record.</td>
</tr>
<tr>
<td>Field</td>
<td>Format</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Given Name</td>
<td>String with a maximum length of 100 characters</td>
<td>Given name or first name for the point of contact provided by the customer. Always appears in records with operation type &quot;Update&quot; or &quot;Create.&quot; Does not appear in records with operation type &quot;Delete.&quot;</td>
</tr>
<tr>
<td>Surname</td>
<td>String with a maximum length of 100 characters</td>
<td>Surname (family name or last name) for the point of contact provided by the customer. Always appears in records with operation type &quot;Update&quot; or &quot;Create.&quot; Does not appear in records with operation type &quot;Delete.&quot;</td>
</tr>
<tr>
<td>Telephone Number</td>
<td>String with a maximum length of 25 characters. May include international phone numbers.</td>
<td>Telephone number provided by the customer. Always appears in records with operation type &quot;Update&quot; or &quot;Create.&quot; Does not appear in records with operation type &quot;Delete.&quot;</td>
</tr>
<tr>
<td>Email</td>
<td>String with a maximum length of 254 characters</td>
<td>Email address provided by the customer. Always appears in records with operation type &quot;Update&quot; or &quot;Create.&quot; Does not appear in records with operation type &quot;Delete.&quot;</td>
</tr>
<tr>
<td>Title</td>
<td>String with a maximum length of 255 characters</td>
<td>Job title provided by the customer. Optional field. Will sometimes occur in records with operation type &quot;Update&quot; or &quot;Create.&quot; Does not appear in records with operation type &quot;Delete.&quot;</td>
</tr>
<tr>
<td>Country Code</td>
<td>2-character ISO 3166 country code</td>
<td>Country code provided by the customer. Optional field. Will sometimes occur in records with operation type &quot;Update&quot; or &quot;Create.&quot; Does not appear in records with operation type &quot;Delete.&quot;</td>
</tr>
<tr>
<td>Field</td>
<td>Format</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ZIP Code</td>
<td>5-digit string</td>
<td>Zip code provided by the customer; applicable to USA only. Optional field. Will sometimes occur in records with operation type “Update” or “Create.” Does not appear in records with operation type “Delete.”</td>
</tr>
<tr>
<td>Operation Time</td>
<td>ISO 8601 date/time, with UTC time zone. The format is YYYY-MM-DDTHH:mm:ss.nnnZ (YYYY is year, MM is month, DD is day of month, HH is hour of day from 00-23, mm is minute of hour from 00-59, ss is second of minute from 00-59 and nnn is millisecond of second from 000-9999), such as “2016-04-07T14:05:15.275Z”</td>
<td>Indicates the date/time when the record was most recently created, updated, or deleted by the customer. Required field; always appears in every record.</td>
</tr>
<tr>
<td>Operation Type</td>
<td>String; possible values are “CREATE”, “UPDATE”, or “DELETE”</td>
<td>CREATE: Indicates that the record has been newly created since the from-date specified in the API request. UPDATE: Indicates that the record has been updated since the from-date specified in the API request. DELETE: Indicates that the record has been deleted since the from-date specified in the API request. Required field; always appears in every record.</td>
</tr>
</tbody>
</table>

An example of the output file format is shown below.

```
Product Id,Product Code,Customer Guid,Subscription Guid,Subscription Start Date,Organization,AWS Customer Id,Given Name,Surname,Telephone Number,Email,Title,Country Code,ZIP Code,Operation Time,Operation Type
4b898955-84fa-4cfa-8f43-98287ad69c06,4gzp2symm0v9zdfrmn9f854w6,ba1d75cc-d984-4f07-bb14-a04b952afbc,cad371fb-6f2c-4537-a054-1a7afca6312fd,2016-05-27T00:00:00.000Z,Example Inc -- Service Division,00001112222,Example Inc,000-947-8228,eugethi@example.org,,,,2016-05-12T03:54:46.143Z,CREATE
4b898955-84fa-4cfa-8f43-98287ad69c06,4gzp2symm0v9zdfrmn9f854w6,1b4a2b5f-2c5d-4779-b0c7-2878b0f45cfc,cad371fb-6f2c-4537-a054-1a7afca6312fd,2016-05-19T00:00:00.000Z,Example Inc -- Service Division,00001112222,Example Inc,000-947-8228,eugethi@example.org,,,,2016-05-19T18:21:06.834Z,CREATE
```
When a customer deletes their contact information from the PSC program, you will see a record in the output .csv file that indicates an operation type "DELETE." After a customer deletes their data, the API no longer transmits contact information such as name, telephone number, email, and so forth. Each delete record consists of the data required to uniquely identify the record to be deleted. Delete records contain product ID, product code, operation time, customer GUID, subscription GUID, subscription start date, AWS Customer ID, operation time, and operation type.

If a customer opts out of Product Support Connection by deleting their contact information, you should also remove the contact information from your records. Because the customer contact data will not be included in the DELETE record, you will need to look up the record in your system by using the unique Customer GUID.

A delete record will also be sent if a customer terminates a subscription.

Note

If you have questions or would like more information about participating in AWS Marketplace Product Support Connection, contact the AWS Marketplace Seller Operations team.
Preparing your product

Preparing to publish a product on AWS Marketplace includes configuring your package, setting a pricing scheme, determining what categories your product shows under, and adding keywords so your product appears in relevant searches.

The following list describes the ways you can deliver products, how AWS Marketplace buyers find each type of deliverable, and links to procedures for creating each type of deliverable.

- **Amazon Machine Image (AMI)** – You can offer AMI-based products in the following ways:
  - As a single AMI.
    
    Buyers can find these products using the **Amazon Machine Image** delivery method filter.
    
    For more information, see [AMI-based products](#) (p. 61).
  - As AMIs delivered using AWS CloudFormation templates.
    
    Buyers can find these products using the **CloudFormation** delivery method filter.
    
    For more information about delivering AMIs as a AWS CloudFormation template, see [AMI-based delivery using AWS CloudFormation](#) (p. 70). For more information about AWS CloudFormation templates, see [AWS CloudFormation concepts](#) in the [AWS CloudFormation User Guide](#).
  - As a private image build. With this method, you offer products in a way that lets buyers install your product on a base gold image that meets their internal standards for operating system configuration.
    
    Buyers can find these products using the **Private Amazon Machine Image** delivery method filter.
    
    For more information, see [Private images](#) (p. 80).

- **Container** – You can deliver products in Docker containers. Container products consist of fulfillment options, which are a set of container images and deployment templates that go together.

    Buyers can find these products using the **Container** delivery method filter.
    
    For more information, see [Getting started with container products](#) (p. 85).

- **File-based data sets** – To deliver file-based data sets, you use AWS Data Exchange, a separate AWS service.

    Buyers can find these products using the **AWS Data Exchange** delivery method filter.
    
    For information about publishing and managing data products and offers through AWS Data Exchange, see [Providing Data Products on AWS Data Exchange](#) in the [AWS Data Exchange User Guide](#).

- **Machine learning algorithms and model packages** – With this method, you use SageMaker a separate AWS service, to create the algorithm or model package, and then publish it on AWS Marketplace.

    Buyers can find these products using the **SageMaker** delivery method filter.
    
    For more information on delivering machine learning algorithms and model packages, see [Putting your algorithms and model packages on the AWS Marketplace](#) (p. 107). For information on SageMaker, see [What is SageMaker?](#) in the [Amazon SageMaker Developer Guide](#).

- **Software as a service (SaaS)** – You can offer SaaS products with subscription-based or contract-based pricing models.

    Buyers find these products using the **SaaS** delivery method filter.
    
    For more information, see [Software as a service (SaaS)–based products](#) (p. 113).
Each delivery method has several options for packaging, pricing, and delivery. Some methods are not available to you as a seller on AWS Marketplace until you register for the program supporting it. You can create products with a standard list price and end user license agreement (EULA), and you can create private offers for individual customers with custom pricing and EULAs. If you need to make additional changes to the terms of the contract, you can work with the AWS Marketplace team to create a custom private offer.

**Tip**

To simplify the procurement process, you can use standardized license terms (p. 52) for both public product listings and private offers.

**Topics**

- Product pricing (p. 30)
- Regions and countries for your AWS Marketplace product (p. 43)
- Private offers (p. 43)
- Standardized license terms (p. 52)
- Categories and metadata (p. 54)
- Search engine optimization (p. 56)

---

**Product pricing**

The following is general pricing information about products in AWS Marketplace. All pricing is based on US dollars (USD). For information on refunds, see Refunds (p. 41).

- For Paid products, AWS Marketplace collects software charges from the customer.
- There is no service fee for Bring Your Own License (BYOL) products on AWS Marketplace.
- To deliver on our customer promise of selection, we require that all BYOL products also have a paid option. This is so that customers who don't have existing licenses have the option to purchase and use the products.
- For BYOL products, we realize that the online purchase of software is a departure from how some companies do business. In light of this, for the first 90 days after launch we will relax the requirement that this software is accompanied by a version available for purchase on AWS Marketplace. During this time, the AWS Marketplace account management teams will work with you to address challenges and to determine if and how the software can be made available for purchase on AWS Marketplace.
- There is no service fee for Free or Open Source Software that is made available to customers without charge.

**AWS charges versus software charges**

- All AMI-based products will incur associated AWS infrastructure charges depending on the services and infrastructure used. These rates and fees are defined and controlled by AWS, and can vary between regions. For more information, see Amazon EC2 Pricing.
- For Paid products, the seller defines the charges for using the software.

These two types of prices are displayed separately on the AWS Marketplace detail pages to help customers understand the potential cost of using the products.

**Free trial**

Hourly products are eligible for the optional Free Trial program, where a customer can subscribe to the product and use a single instance for up to 31 days without paying software charges on the product.
Applicable AWS infrastructure charges still apply. Simply define the duration of the trial period (5 to 31 days) and notify the AWS Marketplace Managed Catalog Operations (MCO) team.

When customers subscribe to a Free Trial product, they receive a welcome email that includes the term of the Free Trial, a calculated expiration date, and details on unsubscribing. A reminder email is sent three days before the expiration date.

If you offer a Free Trial product in AWS Marketplace, you agree to the specific refund policies described under Refund Policy.

Changing prices

You can update prices and metadata through the AWS Marketplace Management Portal.

To change prices

2. In the Products tab, you will find a list of current products that you created. In the table for your current products, choose the Action column to edit your product.

Changing pricing models

Changes to pricing models must be reviewed and approved by AWS Marketplace to ensure a positive customer experience and reduced risk to all parties. Discuss the pricing model changes you want to make by contacting the AWS Marketplace Managed Catalog Operations (MCO) team. All requests for pricing model changes can take 30-90 days to process and review.

Annual pricing

An annual pricing model enables you to offer products to customers who can purchase a 12-month subscription. The subscription pricing can provide up to 40% savings versus running the same product hourly for extended periods. The customer is invoiced for the full amount of the contract at the time of subscription. For more information about how annual subscriptions are presented to customers, see AMI Subscriptions.

See the following considerations when working with an annual subscriptions.

- Annual pricing is defined per instance type. It can be the same for all Amazon EC2 instance types or different for each instance type.
- All Annual instance types must also have an Hourly instance type defined. AWS Marketplace doesn't offer Annual-only pricing or Hourly without Annual on the same product. For any product offering Annual pricing, Hourly pricing also needs to be specified.
- A $0 Annual price is allowed on a specific instance type, if the Hourly price is also $0 and there are other non-$0 Annual instance types defined.
- At the end of the annual subscription period, the customer will start being charged at the hourly price.
- If a customer buys X Annual subscriptions but is running Y software on Y instances, then the customer will be charged at Hourly software price for (Y-X) instances which are not covered by Annual subscriptions. As such, an Hourly rate must be included for all Annual pricing instance types.
- Using seller private offers, you can offer a multi-year (up to 3 years) or custom duration AMI with upfront payment, or flexible payment schedule. For more information about multi-year and custom duration contracts, see Private offers (p. 43) and the section called "Flexible payment scheduler" (p. 46).

If you offer an Annual product in AWS Marketplace, you agree to the specific refund policies for Annual products, located in the File Uploader documents section in the AWS Marketplace Management Portal.
Usage pricing

The AWS Marketplace Metering Service enables you to define additional dimensions you want to charge your customers for the value your software provides. As a seller, you can choose one of the usage categories from the following:

1. Users
2. Hosts
3. Bandwidth
4. Data

You can also define up to 24 dimensions for the product. All charges must be measured and reported every hour from the software deployed in the customer's account. All usage is calculated monthly and billed monthly using the same mechanism as existing AWS Marketplace software.

Using the AWS Marketplace Metering Service, you can handle several new pricing scenarios. For example, if your software monitors hosts, you can charge for each host monitored and set different pricing based on the host size. If your software allows multiple users across an organization, you can charge by user. Each hour, the customer would be charged for the total number of provisioned users.

Note

In the product load form, relevant columns are named as Flexible Consumption Pricing (FCP).

For AWS Marketplace Metering Service products, note the following:

- If your software is already on AWS Marketplace, you will need to create a product to enable an alternate usage dimension. That is, currently, we are unable to convert a standard product to use the AWS Marketplace Metering Service. After the new product is published, you can remove the old product or keep both on site.
- The AWS Marketplace Metering Service requires that your software reports usage every hour, recording the customer usage for the hour. If there is a failure in the transmission or receipt of metering service records, AWS will be unable to bill for such usage. You are responsible for ensuring the successful receipt of metering records.
- At this time, products that use the AWS Marketplace Metering Service don't support 1-Click. Buyers are required to launch your software with an IAM role with specific permissions and have an Internet Gateway.
- Free Trial and Annual Pricing are not compatible with the AWS Marketplace Metering Service at this time.
- Changing dimension (user, hosts, bandwidth, and data) or dimension name is not supported. You will need to create a new product.

Private offers

The AWS Marketplace Seller Private Offer program allows AWS Marketplace sellers to negotiate custom pricing and end user license agreements with individual AWS Marketplace customers (buyers). For more information, see Private offers (p. 43).

SaaS subscriptions pricing

For SaaS Subscriptions, AWS Marketplace bills your customers based on the metering records received by us. All charges must be measured and reported every hour from the software deployed in the customer's account. All usage is then calculated monthly and billed monthly using the same mechanism as AMI based AWS Marketplace offerings. AWS' ability to bill customers for usage of your product is dependent
upon receiving metering records from you. You are responsible for ensuring that your product’s metering records are successfully transmitted and received.

SaaS contracts pricing

For SaaS Contracts, the customer initiates a purchase of your software and enters into an agreement with you. Under the agreement, the customer is entitled to a specified quantity of use of your SaaS product. AWS Marketplace communicates these entitlements to your SaaS application. This is done through the AWS Marketplace Entitlement Service. When using SaaS Contracts, your application never sends metering records. Instead, it verifies entitlement by calling the AWS Marketplace Entitlement Service. You define the usage categories, dimensions, and the length of the contract.

AMI pricing models

AWS Marketplace has multiple pricing models for AMI products. With seller private offers, there are options available for multi-year and custom duration contracts. For more information about multi-year and custom duration contracts, see Private offers (p. 43) and the section called “Flexible payment scheduler” (p. 46). The following table provides general information about pricing models.

Note
You must be able to provide a W-9 tax form (for U.S. based entities) or a W-8 form (for EU-based entities) as described in the section called “Seller registration process” (p. 5).

<table>
<thead>
<tr>
<th>Pricing model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bring Your Own License (BYOL)</td>
<td>AWS Marketplace does not charge customers for usage of the software, but customers must supply a license key to activate the product. This key is purchased outside of AWS Marketplace. The entitlement/licensing enforcement, as well as all pricing and billing are handled by you.</td>
</tr>
<tr>
<td>Free</td>
<td>Customers are allowed to run as many instances as Amazon EC2 supports with no additional software charges incurred.</td>
</tr>
</tbody>
</table>
| Hourly                        | **Hourly** – Software is charged by the hour. Each instance type can be priced differently (but is not required to be) and usage is rounded up to the nearest whole hour.  

**Hourly with Free Trial** – Customers are limited to running exactly one instance of the software without incurring a charge. You define the duration, between 5 and 30 days. The free trial applies to the most expensive instance type that is running, and any concurrent usage outside the 1 instance is billed at the hourly rate. NOTE- This is a different model than the AWS Free Tier for Amazon EC2 usage whereby customers are given 750 hours of free usage each month.

**Hourly with Monthly** – Both hourly and monthly charges are applied independently; the monthly fee is charged every month regardless of usage, the hourly fee is applied based on hourly usage only. |
<table>
<thead>
<tr>
<th>Pricing model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hourly with Annual</strong></td>
<td>Customers have the option to purchase a year's worth of usage upfront for one Amazon EC2 instance of one instance type. You set the pricing for each instance type and can offer net savings over the hourly price. Any customer usage above the number of annual subscriptions purchased is billed at the hourly rate you set for that instance type.</td>
</tr>
<tr>
<td><strong>Hourly with Multi-Annual and Custom Duration</strong></td>
<td>This type offer is only available through seller private offers. Using seller private offers, you specify a custom contract duration, up to 3 years. You can specify upfront payment, or include a flexible payment schedule. You set the pricing for each instance type. If there is a flexible payment schedule in the offer, you also set the invoice dates, payment amounts, and number of instances for each instance type included in the offer. For an active seller private offer with a flexible payment schedule, after the customer launches the specified number of instances, any additional instances launched are charged at the hourly rate specified in the seller private offer. For more information about multi-year and custom duration contracts, see Private offers (p. 43) and the section called “Flexible payment scheduler” (p. 46).</td>
</tr>
<tr>
<td><strong>Hourly with Free Trial and Annual</strong></td>
<td>This is identical to the Hourly model with an Annual option, except it includes a Free Trial allowing a customer to run 1 instance of any instance type for free for a set number of days that you determine. Annual subscriptions can be purchased at any time, and they are combined with the Free Trial subscription.</td>
</tr>
<tr>
<td><strong>Monthly</strong></td>
<td>Software is paid for on a fixed monthly basis, regardless of the number of instances the customer runs. Monthly Charges are pro-rated at sign-up and upon cancellation. Example: A customer who subscribes for 1 day of the month will be charged for 1/30th of the month.</td>
</tr>
<tr>
<td><strong>Monthly with Hourly</strong></td>
<td>Both Hourly and Monthly charges are applied independently. The monthly fee is charged every month regardless of usage and the hourly fee is applied based on hourly usage only.</td>
</tr>
<tr>
<td><strong>Note</strong></td>
<td>Free Trial and Annual pricing cannot be combined with Monthly pricing.</td>
</tr>
</tbody>
</table>
### Pricing your software with SaaS

To set prices, you first define pricing dimensions that represent the units of value in your software and then assign a price to each dimension. For example, dimensions can be protected hosts, users, or storage volumes. You can define up to 24 dimensions. Next you will also select a category for those dimensions which can be one of our preset categories (bandwidth, data, hosts, requests, tiers, users). If none of

<table>
<thead>
<tr>
<th>Pricing model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Annual</strong></td>
<td><strong>Annual with Hourly</strong> – Same as the hourly with annual pricing model. Customers have the option to purchase a year’s worth of usage upfront for one Amazon EC2 instance of one instance type. You set the pricing for each instance type and can offer net savings over the hourly price, but offering savings is not required. Any customer usage above the number of annual subscriptions purchased is billed at the hourly rate you set for that instance type.</td>
</tr>
<tr>
<td><strong>Multi-Annual and Custom Duration with Hourly</strong> – This is only available through Private offers (p. 43). Using seller private offers, you can specify a custom duration contract of up to three years. You can require upfront payment, or can offer a flexible payment schedule to the customer. You set the pricing for each instance type for the duration of the contract, and the hourly pricing for additional instances launched. If you offer a flexible payment schedule, you also set the invoice dates, payment amounts, and number of instances for each instance type included in the offer. For an active private offer with a flexible payment schedule, after the specified number of instances have been launched, any additional instances the customer launches are charged at the hourly rate specified in the private offer. For more information about multi-year and custom duration contracts, see Private offers (p. 43) and the section called “Flexible payment scheduler” (p. 46).</td>
<td></td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td><strong>Usage</strong> – Software is directly charged for the value you provide along with one of four usage categories: users, data, bandwidth, or hosts. You can define up to 24 dimensions for the product. All charges are still incurred hourly by the customer. All usage is calculated monthly and billed monthly using the same mechanism as existing AWS Marketplace software. Usage pricing is also referred to as AWS Marketplace Metering Service</td>
</tr>
</tbody>
</table>

**Note**

Free trial and Annual pricing cannot be combined with Usage pricing.
the presets fit your use case, you can choose the generic 'units' category and describe the units in the
dimension description.

Example: Provisioned bandwidth with nonlinear pricing

Imagine you offer network appliance software. You choose to bill by provisioned bandwidth. For your
usage category, select bandwidth. In addition to charging by bandwidth, you want to charge a different
price as buyers scale up. You can define multiple dimensions within the bandwidth category. You can
define a distinct price for 25 Mbps, 100 Mbps, and 1 Gbps.

Example: Concurrent hosts with multiple dimensions

Imagine you offer software that monitors other Amazon EC2 instances. You choose to bill by the number
of hosts that are being monitored. For your usage category, select host. In addition to charging by host,
you want to charge for the extra value for monitoring larger hosts. You can use multiple dimensions
within the host category. You can define a distinct price for micro, small, medium, large, x-large, 2XL,
4XL, 8XL instances. Your software is responsible for mapping each particular host to one of your defined
dimensions. Your software is responsible for sending a separate metering record for each dimension of
your usage category if applicable.

Listing your SaaS product on AWS Marketplace

To take advantage of the Metering Service, you must create a new product. If your product is already on
AWS Marketplace, you will need to decide whether the new AWS Marketplace Metering Service product
will be made available in addition to your current one, or if it will replace it as the only version available
to new users. If you choose replacement, the existing product will be removed from the AWS Marketplace
so that it is no longer available for new buyers. Existing customers will continue to have access to their
old product and instances, but they can migrate to the new product at their convenience. The new
product must meter usage to the AWS Marketplace Metering Service.

After you have your AMI, follow the standard process to share and scan your AMI using the self-service
tool. In addition, using the template available on the management portal, fill out the product load form
and upload it to start the ingestion process.

The following definitions will help you fill out the fields of the product load form for the AWS
Marketplace Metering Service. On the product load form, these fields are labeled as Flexible
Consumption Pricing (FCP) to differentiate them from hourly and monthly priced products.

- **Title** – If you already have a product, and you are adding the same product with the AWS Marketplace
  Metering Service, include the FCP category/dimension in parenthesis to differentiate the two (for
  example, “PRODUCT TITLE (Data)”)
- **Pricing Model** – From the dropdown list, choose Usage.
- **FCP Category** – The category in which customers will be charged for paid products with a Usage
  pricing component. From the dropdown menu, choose Users, Hosts, Data, or Bandwidth.
- **FCP Unit** – The unit of measurement on which customers will be charged for paid products with a
  Usage pricing component. Options will appear in the dropdown menu based on the FCP category that
  you chose.

The following table lists the valid units for each category.

<table>
<thead>
<tr>
<th>Category</th>
<th>Valid units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>UserHrs</td>
</tr>
<tr>
<td>Hosts</td>
<td>HostHrs</td>
</tr>
<tr>
<td>Data</td>
<td>MB, GB, TB</td>
</tr>
</tbody>
</table>
### Pricing your software with SaaS

<table>
<thead>
<tr>
<th>Category</th>
<th>Valid units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth</td>
<td>Mbps, Gbps</td>
</tr>
</tbody>
</table>

- **FCP Dimension Name** – The name used when sending metering records by calling MeterUsage API. It is visible in billing reports, but because it is not external-facing, the name does not need to be user-friendly. The name can be no more than 15 characters and can only include alphanumeric and underscore characters. After you set the name, you will not be able to change it. Changing the name requires a new AMI.

- **FCP Dimension Description** – The customer-facing statement that describes the dimension for the product. The description (for example, Administrators per hour, Per Mbps bandwidth provisioned) can be no more than 70 characters and should be user-friendly. After the product is published, you will not be able to change this description.

- **FCP Rate** – The software charge per unit for this product. This field supports 3 decimal places.

**Note**

- You don't need to fill out hourly and annual pricing fields.
- Free trial and annual pricing are not compatible.
- Products that use the clusters and AWS Resources feature cannot use the AWS Marketplace Metering Service.
- Price, instance type, or region change will follow the regular process as other AWS Marketplace products.
- Products with the AWS Marketplace Metering Service cannot be converted to other pricing models such as hourly, monthly, or BYOL.
- We recommend adding IAM policy information in your usage instructions or document.

If you have questions, contact the [AWS Marketplace Managed Catalog Operations (MCO)](https://aws.amazon.com/marketplace/)

### Modifying your SaaS software to use the Metering Service

You will need to modify your software to record customer usage, send hourly usage reports to the Metering Service, and handle new failure modes. The software operates independently of pricing, but the software will need to know about the usage category, how it is consumed, and any dimensions.

### Measuring consumption

Your software must determine how much of the selected usage category and which dimensions the customer has consumed. This value will be sent, once each hour, to the AWS Marketplace Metering Service. In all cases, it is assumed that your software has the ability to measure, record, and read consumption of resources for the purpose of sending it on an hourly basis to the Metering Service.

For provisioned consumption, this will typically be read from the software configuration as a sampled value, but might also be a maximum configured value, recorded each hour. For concurrent consumption, this might be either a periodic sample or a maximum value recorded each hour. For accumulated consumption, this will be a value that is accumulated each hour.

For pricing on multiple dimensions, multiple values must be measured and sent to the Metering Service, one per dimension. This requires your software to be programmed or configured with the known set of dimensions when providing the product AMI. The set of dimensions cannot change after a product has been created.

For each pricing scenario, this table describes recommended ways for measuring consumption each hour:
<table>
<thead>
<tr>
<th>Scenario</th>
<th>How to measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provisioned User</td>
<td>Current number of provisioned users (sampled).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum number of provisioned users (seen that hour).</td>
</tr>
<tr>
<td>Concurrent User</td>
<td>Current number of concurrent users (sampled).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum number of concurrent users (seen that hour).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Total number of distinct users (seen that hour).</td>
</tr>
<tr>
<td>Provisioned Host</td>
<td>Current number of provisioned hosts (sampled).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum number of provisioned hosts (seen that hour).</td>
</tr>
<tr>
<td>Concurrent Host</td>
<td>Current number of concurrent hosts (sampled).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum number of concurrent hosts (seen that hour).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Total number of distinct hosts (seen that hour).</td>
</tr>
<tr>
<td>Provisioned Bandwidth</td>
<td>Current provisioned bandwidth setting (sampled).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum provisioned bandwidth (seen that hour).</td>
</tr>
<tr>
<td>Accumulated Data</td>
<td>Current GB of data stored (sampled).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum GB of data stored (seen that hour).</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Total GB of data added or processed that hour.</td>
</tr>
<tr>
<td></td>
<td>- OR-</td>
</tr>
<tr>
<td></td>
<td>Total GB of data processed that hour.</td>
</tr>
</tbody>
</table>
Call AWS Marketplace Metering Service

Your software must call the Metering Service hourly and record the consumption value for that hour.

When your software starts, it should record the minute-of-the-hour at which it started. This will be referred to as the start-minute. Every hour on the start-minute, your software must retrieve the consumption value for that hour and call the Metering Service.

To wake up each hour at the start-minute, your software will need to use one of three approaches:

1. A thread within your software.
2. A daemon process that starts up with the instance or software.
3. A cron job that is configured during application startup.

Your software must call the AWS Marketplace Metering Service using the IAM role configured on the customer's instance and specify the consumption dimension and amount.

Your software can use the AWS SDK to call the AWS Marketplace Metering Service. The following is a typical implementation:

1. Use the instance profile to create a service client. This requires the role configured for the Amazon EC2 instance. The role credentials are refreshed by the SDK automatically.

   Example

   ```java
   AmazonMeteringService meteringClient = new AmazonMeteringService(new
   InstanceProfileCredentialsProvider());
   ```

2. Each hour, read your software configuration and state to determine consumption values for that hour. This might include collecting a value-per-dimension.

3. Call the `meterUsage` action on the SDK client with the following parameters (call additionally for each dimension that has usage):
   - `timestamp` – Timestamp of the hour being recorded (use UTC).
   - `productCode` – The product code assigned to the software.
   - `dimension` – The dimensions assigned to the software.
   - `quantity` – The consumption value for the hour.

In addition, your software must call an in-region AWS Marketplace Metering Service endpoint. Your product must have a correct regional endpoint setup, so US East (N. Virginia) sends records to US East (N. Virginia) endpoint, and US West (Oregon) sends records to US West (Oregon) endpoint. Making in-region calls provides buyers with a more stable experience and prevents situations in which an unrelated region's availability could impact software running in another region.

When you send metering records to the service, you must connect to the AWS Marketplace Metering Service in your region. Use the `getCurrentRegion` action to determine the region in which the Amazon EC2 instance is running, and then pass this region information to the `MeteringServiceClient` constructor. If you don't specify a region in the SDK constructor, it defaults to the us-east-1 region. If your application attempts to make cross-region calls to the service, it will be rejected.

Failure handling

Your product must send metering records to the service, a public internet endpoint, so that usage can be captured and billed. Because it's possible for a customer to modify network settings in a way that prevents your metering records from being delivered, your product should account for this by choosing a failure mode.
Typically, software can fail open (provide a warning message but maintain full functionality) or fail closed (disable all functionality in the application until a connection has been reestablished). You can choose to fail open, closed, or something specific to your application. We recommend that you refrain from failing closed after less than two hours of metering failures.

As an example of failing partially open, you could continue to allow access to the software but not allow the buyer to modify the software settings. Or, a buyer could still access the software, but would not be able to create additional users. Your software is responsible for defining and enforcing this failure mode. Your software’s failure mode must be included when your AMI is submitted, and it can’t be changed later.

**Annual products**

These guidelines apply to all sellers who are offering a product on AWS Marketplace with annual pricing.

**Price change**

You can change annual prices (the $ value, for example $1000/year to $1200/year) whenever desired but with 90 day notice to existing customers of annual pricing. The new price will apply to new subscriptions but will have no impact on existing subscriptions. Price changes will be effective for auto-renewals only if the price was changed at least 90 days before the auto-renewal date. The customer will receive an email prior to auto-renewal that includes the new price.

**Refund / cancellation / upgrade / downgrade**

For uniform and great customer experience, AWS requires sellers to implement the following cancellation/change windows:

<table>
<thead>
<tr>
<th>Applicable policy</th>
<th>Time period or window</th>
<th>Who can authorize it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full refund cancellation (cancel with 100% refund)</td>
<td>Within 48 hours of purchase</td>
<td>• Public purchases – AWS customer support or the seller</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Private offers – seller only</td>
</tr>
<tr>
<td>Pro-rata refund cancellation (cancel with pro-rata refund)</td>
<td>Within 14 days of purchase</td>
<td>Seller only</td>
</tr>
<tr>
<td>Downgrade subscription (replace existing subscriptions with less expensive subscription)</td>
<td>Within 30 days of purchase</td>
<td>Seller only</td>
</tr>
<tr>
<td>Upgrade subscription (replace existing subscriptions with more expensive or same priced subscription)</td>
<td>Any time during 12 months</td>
<td>• Public purchases – AWS customer support or the seller</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Private offers – seller only</td>
</tr>
<tr>
<td>Full refund cancellation in case of auto-renewal</td>
<td>Within 14 days of purchase</td>
<td>AWS customer support or the seller</td>
</tr>
</tbody>
</table>

**Note**

- You should not include windows length and other details in product details/description.
- Upgrade or downgrade is a 2-step process for customer: buy new subscriptions and request cancellation of old subscription with a refund.
- In some cases AWS might issue refunds on your behalf. No action on your part is required to process those refunds.
End user license agreement

AWS customer’s usage of software for 12 months under annual subscription is covered by the EULA you have provided on your product’s details page on AWS Marketplace.

Refunds

All paid products, regardless of pricing model, must have a stated refund policy for software charges. The policy must include the terms of the refund as well as a method of contacting the seller to request a refund. While the details of the refund policy are up to you, we encourage you to offer customers some manner of refund for usage of the product. You must comply with your posted refund policies.

Refund request types

Customers can request different kinds of refunds for AWS Marketplace products. If a customer requests a software refund directly from AWS, we instruct them to contact you using your posted support contact information for the product in question. Refunds of any AWS infrastructure charges are up to the discretion of AWS and are handled independently of software refunds.

If you use the AWS Marketplace Tax Calculation Service, customers might contact you to request a tax-only refund. If a customer requests a tax-only refund, you can, at your discretion, grant either a tax-only refund or a full software refund plus tax.

Refund approvals

Most software refunds must be approved by you through the Refund Request Form, and will be processed and issued upon your direction by AWS. Special cases where your approval is not needed exist for the following product types:

- Pro-rated subscription cancellations
- Usage-based pricing
- Free trials

In these cases you agree that AWS can initiate and process refunds without approval or direction from you. These kinds of refunds are processed automatically and require no action on your part. For more information, see the following related topics:

- Annual products (p. 40)
- Refund policy (p. 42)

Refund process

You can initiate refunds for your product software usage by submitting a Refund Request Form. Once received by the AWS Marketplace Buyer Support Team, a related support case will be created in the AWS Support Center Console, with the refund status noted in the subject line. Refund related support is facilitated directly through these cases. For more information, see Accessing AWS Support.

The following procedure outlines how to request a refund. Submitting a refund request for an internal testing account follows the same steps as external customers.

To initiate a software refund for a customer

1. Gather the following information from the customer:
Refunds

- The customer’s email address that is associated with their AWS account.
- The customer’s AWS account number of the account used to subscribe to your product. Remind your customer that if they are the payer of an organization, they need to provide you with the AWS account ID for the linked account subscribed to your product.
- The billing period(s) for which the customer would like a refund.

2. Navigate to the Refund Request Form.
3. Write the customer’s information in the form.
4. Enter the Product ID for the product requesting a refund for. You can find the Product ID in your daily Subscription Report.
5. For annual products where a customer is requesting a refund, upgrade or downgrade, you must perform the following tasks:
   a. Verify the customer has purchased an annual subscription using your Subscriber Report (there might be a 24-hour delay).
   b. Provide a Subscription Cancellation Date in the comments field.
   c. Provide a description of the change that you’re authorizing (refund, upgrade, or downgrade) in the comments field.
6. Submit the form. We’ll be notified and will begin to process the refund and issue it to the customer.
7. An outbound case will be created in the AWS Support Center Console with status information on the refund request. The subject line will contain one of the following:
   - Completed – The refund was processed and no further action is required.
   - Pending – The refund will be processed once the current billing cycle ends.
   - Action Required – The request could not be processed, and we need additional information from you. You can respond directly to the support case, however you will also need to submit a new refund request form.
8. Once a refund is successfully processed, it will reflect on the customer’s account within 24-48 hours. However, it can take up to 5 business days for the funds to appear in the customer’s financial account.

Refund policy

The following list outlines the AWS Marketplace refund policy:

- If you list your software as a free trial product, AWS can issue refunds on your behalf for software charges accruing within seven days of a conversion from a free trial to a paid subscription. Refunds issued in connection with free trial conversions require no action on your part. By enabling a free trial on a product, you are agreeing to this policy.
- If you meter the usage of your software using the AWS Marketplace Metering Service, AWS can issue refunds on your behalf for software charges resulting from software metering errors. If these errors are common across multiple customers, AWS reserves the right to determine an appropriate refund for each customer and apply it directly to each customer. Refunds issued in connection with the AWS Marketplace Metering Service must be confirmed with the seller once, but does not require the seller to confirm each individual refund. By using the AWS Marketplace Metering Service with a product, you are agreeing to this policy.
- All refunds for private offers must be authorized by the respective seller before AWS can process them.

42
Regions and countries for your AWS Marketplace product

When you create a product in AWS Marketplace, you choose the AWS Regions where it is available. You also choose the countries where buyers can purchase your product from. These two properties are similar, but they are not the same. For example, a buyer might be located in, and purchasing from, the United States but is installing your product in the Europe (Frankfurt) Region. In order for this buyer to purchase your product, you must include both the United States in your list of countries, and the Europe (Frankfurt) Region in your list of Regions.

AWS Regions

When creating or editing server or machine learning product information, you can limit your product to specific AWS Regions where your users can install and use the product.

For server products, including Amazon Machine Image (AMI)-, container-, and AWS CloudFormation-based products, you can select specific Regions where the product is available. You can also choose to automatically make your product available in new US Regions, non-US Regions, or all Regions as they become available.

For machine learning products, you can either select specific Regions, or all Regions including future Regions as they become available.

For more information about AWS Regions, see AWS service endpoints in the AWS General Reference.

Countries

By default, your product is available to buyers in all countries where AWS Marketplace is available. For new and existing server and software-as-a-service (SaaS) products, you can control product availability in specific countries for tax, compliance, support or marketing purposes.

There are exceptions to this functionality:

- **Previous purchases** – After updating your product with a new list of countries, buyers that have already subscribed to your product will still have access while their subscription is active.

- **Private offers** – When you limit your product to buyers in specific countries, it does not limit private offers. When you create a private offer to a specific buyer, it is available to that buyer, even if they are in a country that you did not include in your specified countries.

  **Note**
  Customer eligibility is determined at an AWS linked account level. For more information, see How does AWS determine the Location of your account?

Private offers

Private offers are a purchasing program that allows sellers and buyers to negotiate custom prices and end user licensing agreement (EULA) terms for software purchases in AWS Marketplace.

**Tip**
You can negotiate EULA terms for each private offer, or you can use or amend standardized license terms (p. 52) to simplify the procurement process.
How private offers work

You can create and manage all of your private offers from the Offers page in the AWS Marketplace Management Portal. You specify the product that the offer is being made for and the AWS account ID (or IDs) for the buyer you're creating the offer for. AWS Marketplace Management Portal generates a unique ID and URL for the offer. For instructions on creating private offers, see Consulting partner creates.

When you create a private offer, you can extend the offer to up to 25 accounts. The offer is visible only to the accounts that you create the offer for. Buyers can't view the offer unless you extend the offer to either their linked account or to their management account. You can't force service limits in the offer, so the buyer can use as much of your product at the negotiated prices as they want, unless the product already has a limit.

AWS Marketplace buyers can access third-party financing for private offers. For more information, see Customer financing is now available in AWS Marketplace.

Note
The buyer isn't notified that you created a private offer. You can provide the URL for the custom offer to the buyer, or they can navigate to your product through AWS Marketplace.

When the buyer navigates to your product's subscription page, a banner indicates that a private offer is available. After the buyer accepts the offer, they're invoiced for the purchase using the same portal tools used for all AWS Marketplace transactions. Accepted offers become agreements, and are also referred to as contracts or subscriptions.

For software as a service (SaaS) contract and SaaS contract with consumption products, you can offer upgrades and renewals on agreements that were made when buyers accepted private offers. For example, you can do this to grant new entitlements, offer pricing discounts, adjust payment schedules, or change the end user license agreement (EULA) to use standardized license terms. For more information, see Private offer upgrades and renewals (p. 50).

Private offers are tracked in seller reports. For more information, see Reporting for private offers (p. 46) and the Seller reports guide.

Private offer experience for buyer

After you create a private offer and notify the potential buyer, they will have steps they must perform to accept the offer. For more information about the buyer experience for private offers, see Private offers in the AWS Marketplace Buyer Guide.

To receive the terms of the offer, the buyer must accept the offer before the offer expiration date. After the offer expires, the terms are no longer valid. You must re-create the private offer for the buyer to accept the terms. As the seller, you can provide a URL to the fulfillment page for the offer, or the customer can navigate to your product page on AWS Marketplace and choose the link in the banner to view the private offer.

Private offers through consulting partners

If you are a consulting partner, you can negotiate special terms with an ISV to offer their products to buyers. With this type of offer, you are listed as seller-of-record.

For more information, see Consulting partner private offers (p. 47).

Notes about private offers

When working with private offers, keep the following in mind:

- You can't create private offers for second party, Amazon Machine Image (AMI) monthly, or multi-AMI-based delivery using AWS CloudFormation products, or for limiting customer usage.
• For private offers with the flexible payment scheduler, it is possible to break upfront commitments into multiple payments over time if buyers are on invoicing terms with AWS.

## Supported product types

Currently, AMI and SaaS products are supported for private offers.

### Private offers for AMI products

You can provide private offers pricing for AMI contracts. The offer can be any custom duration in days, up to 3 years (1,095 days). License entitlements begin on the date the buyer accepts the private offer. For AMI private offers with flexible payment schedules, you can set the number of instances agreed to in the contract, for the duration of the contract. You can also define a custom hourly price for those same instances if the buyer uses more.

### Private offers for SaaS products

SaaS private offer products can't change the pricing level for a given pricing tier based on timing. For example, an offer can't charge $0.80/hour for three months and then change pricing to $0.60/hour thereafter for the same pricing tier. For SaaS contracts, private offers don't monitor usage.

Buyers can manually upgrade to new contracts levels at any time, but it is up to the independent software vendor (ISV) to define contract tiers, enforce service limitations, and advise buyers to manually upgrade to higher contract tiers when needed. The duration dimensions that appear on a SaaS contract private offer matches the durations enabled when the public product listing was created. For SaaS contracts, this can be 1 month, 1 year, 2 years, and/or 3 years.

## Offer submission process

You can create simple private offers using the AWS Marketplace Management Portal using the following procedure.

### To create a private offer

2. On the Manage Private Offer page, choose CREATE AN OFFER.
3. On the Create Private Offer page, select the product from the drop-down list and enter the AWS account ID (or IDs) of the AWS Marketplace buyer. If your buyer is paying for the product in installments, select Allow buyers to pay for this product in installments. Verify the information that you entered and then choose NEXT.
   
   **Note**
   
   Selecting Allow buyers to pay for this product in installments (ISV only) enables you to offer your buyer a payment schedule with annual payments that aren't evenly distributed, multiple payments for a multi-year deal, or quarterly payments. Buyers must be on invoicing terms with AWS to receive a flexible payment schedule on their private offer. For more information, see the section called “Flexible payment scheduler” (p. 46).
4. On the Create an Offer page, verify the product name and buyer ID.
5. If the product offering is for an AMI hourly or AMI annual pricing model, specify the **Contract duration**, by choosing an option button or entering a custom duration in number of days.
   
   **Note**
   
   The duration of the offer can be up to 1,095 days.
6. In **Input offer price**, enter the pricing information that you negotiated with the customer. If you have installment payments for the private offer, specify the number of units and the payment
schedule for the contract duration. For more information about installment payments, see the section called “Flexible payment scheduler” (p. 46).

7. In **Upload End User License Agreement**, select from available options or upload your EULA PDF file.

8. In **Offer Expiration and Acceptance Date**, enter the number of days that the offer is valid for.

   **Note**
   This is the number of days after the customer accepts the offer that the terms of the agreement are active. After the number of days has lapsed, the price and EULA revert to the terms provided in the public offering.

9. For **Buyer needs to accept the offer by**, enter the date when the offer is no longer available if not accepted.

   **Note**
   This is the date that the offer becomes null and void. On that date, the buyer won't be able to accept the offer under the custom terms that you have specified.

10. Choose **REVIEW OFFER**.

11. On the **Review Offer** page, verify the offer information and the PDF file, and then do one of the following:

   - If the offer is correct, choose **EXTEND OFFER**.
   - If the offer is incorrect, choose **EDIT OFFER** and make any required changes.

The offer should appear on the **Manage Private Offer** page in approximately 45 minutes. To view the offer, sign in to the AWS Marketplace Management Portal and choose **Private Offer**. This opens the **Manage Private Offer** landing page.

### Reporting for private offers

Private offers appear on the existing seller reports and in the reports relevant to the offer. The **section called “Monthly billed revenue report”** (p. 168) is generated every month and has offer visibility and offer ID information. When an invoice is generated for a buyer, it appears in the report covering the appropriate billing period. For more information, see the **Seller reports guide**.

The **Offer ID** field contains the unique offer ID generated for the private offer. It's blank unless the report entry is for a private offer. The **Offer Visibility** field indicates whether the report entry is a public or private offer. For all private offers, the entry is marked private.

### Flexible payment scheduler

Flexible payment scheduler enables you to extend private offers with a custom payment schedule. The schedule can be spread over up to three years, and the customer makes payments in regular installments. After they are subscribed, your customers can see all the payments on the schedule and on their AWS invoice, helping them track their spending. Flexible payment scheduler is available for private offers on AMI multi-year and SaaS contracts products.

Any customer on invoice terms, for example net-30 or net-60 terms, can subscribe to a private offer with a flexible payment schedule. Customers who pay their AWS bill using a credit card can't. If you try to create a private offer with a custom payment schedule for a customer who isn't on invoice terms, you receive an error.

### Creating a payment schedule

The process for creating a custom payment schedule using flexible payment scheduler is part of the process for creating a private offer. While creating the private offer, as you are adding product and buyer account information, choose **Allow Buyers to pay for this product in installments**. This enables you to create an offer with a flexible payment schedule. When you choose **Next** to continue, the flexible
payment scheduler feature validates that any AWS account that you added is an account on invoice terms. If you have provided an account that isn't on invoice terms, you receive an error message.

**Note**
If the account is in an AWS Organizations billing family, the targeted account can be any account that is on net payment terms with AWS. For more information, see Consolidated Billing for AWS Organizations in the AWS Billing and Cost Management User Guide.

After the AWS account or accounts are confirmed, customize your offer details on the next page. Choose the contract duration for this offer and specify the offer details accordingly.

**Note**
For private offers with flexible payment scheduler, for multi-year and custom duration Amazon Machine Image (AMI) products, set the number of instances for each instance type included in the offer and the hourly pricing for any additional launched instances. After the customer launches the specified number of instances, any additional instances launched are charged at the hourly rate specified in the private offer.

Under **Payment Schedule**, add the invoice dates and invoice amounts for all of the installments that the customer will make. You can add up to 36 installments. Each time you add an installment, **Total amount due from buyer** is updated.

**Note**
The invoice date for the first installment is the first time that the customer is invoiced for your private offer. You receive the payment for that first invoice after AWS Marketplace receives the payment from the customer.

The flexible payment scheduler feature validates that the invoice dates fall within the contract duration. If your last invoice date is after the duration of the contract, you receive an error message.

After you have added all invoice dates and amounts, confirm that **Total amount due from buyer** matches the total price that you want your customer to pay over the course of the private offer. To finish creating the private offer, upload the end user license agreement (EULA) for the customer and set the offer acceptance date.

**Note**
Only one invoice date can occur before the offer acceptance date that you're extending to your customer.

Your customer is invoiced based on the schedule that you defined, and invoices start after they accept the offer. If the first invoice date is scheduled before the offer is accepted, this invoice is processed immediately after the offer is accepted.

**Note**
You can't modify the payment schedule on a private offer that has been extended to and subscribed by a buyer. To make changes, you must create a new offer.

**Reporting for flexible payment scheduler**

Reporting for private offers with flexible payment schedules is in the Section 4: Contracts with flexible payment schedule (p. 173), of the monthly billed revenue report.

**Consulting partner private offers**

AWS Marketplace consulting partner private offers allow consulting partners to resell independent software vendors' (ISVs) products on AWS Marketplace. The consulting partner and ISV establish an agreement to resell one or more of the ISV's products, and then they extend a private offer to the buyer for that product.

Each consulting partner private offer is visible only to a single buyer, with customized pricing and unique commercial terms to meet that buyer's needs. When creating a private offer, you start from a wholesale
cost set by the ISV. Then you mark up that price to create the buyer’s offer price. The wholesale cost is determined in one of two ways:

- **Recurring discount** – An ISV authorizes the consulting partner to resell their product or products at an agreed-to discount from their list price with a recurring opportunity. This discount allows the consulting partner to continue to resell the product without further price negotiation with the ISV.

- **Non-recurring discount** – The opportunity that the ISV gives the consulting partner is a discount intended to be used only with a specific buyer.

In both cases, after the buyer pays for the private offer, AWS Marketplace uses the standard process to distribute the funds to the consulting partner and the ISV based on the agreed-to pricing.

For detailed instructions about creating private offers, see Consulting partner creates. For information about third-party financing for private offers, see Customer financing is now available in AWS Marketplace.

### Additional information

For additional information and questions, we encourage ISVs and consulting partners to connect with the AWS Marketplace channel team. If you don’t know who to contact specifically, send an email message to <aws-mp-channel@amazon.com>, and someone on the team will respond to you within one business day.

### ISV setup of resell opportunities

As an ISV, you can authorize consulting partners to resell your products by creating a resale opportunity for that partner. You can specify a discount percentage or custom price per product dimension to create a wholesale price for the consulting partner. The partner can mark up the wholesale price when creating their consulting partner private offer for a buyer. For more information about consulting partner private offers, see Extending a private offer based on an opportunity (p. 49).

**Note**

If the particular terms of the authorization that you want to create are not possible using the AWS Marketplace Management Portal, you can fill out an AWS Marketplace Reseller Author form. To request and return the form, reach out to your AWS Marketplace channel account manager or send an email message to mpcustdesk@amazon.com.

The following procedure outlines how ISVs can create an opportunity for a consulting partner. To use this feature, you must have permissions to use the Partners tab in the AWS Marketplace Management Portal. For more information, see Policies for AWS Marketplace sellers (p. 207).

### To create an opportunity for a consulting partner

2. Choose the Partners tab, and then choose Create opportunity.
3. Enter the Opportunity name and Opportunity description, and the Duration of the opportunity.

**Note**

The information you enter in Opportunity name and Opportunity description will be visible to consulting partners in their seller reports.

4. Choose the Resellers that you want to authorize. You can select resellers by name or account ID.

**Note**

If a reseller doesn't appear in the list, they may need to register first. Only registered resellers can be authorized for an opportunity. For more information, see Consulting partner setup of resell opportunities (p. 49).
5. Select which of your **Products** are part of this opportunity, and the **Discount** that you want to apply. Optionally, set one or more **Buyer account IDs** to specify that the opportunity is only for those buyers.

6. Select **Review opportunity**, and make sure that the information is correct.

7. Select **Create opportunity** to finalize the opportunity and authorize the consulting partners.

Once created, opportunities can't have their dates extended. However, you can revoke an opportunity and recreate it at any time. When you revoke an opportunity, new offers can't make use of that discount. Any existing offers are unaffected and retain their opportunity discount.

**Consulting partner setup of resell opportunities**

To create a consulting partner private offer, you must be registered through the AWS Marketplace Management Portal as an AWS Marketplace seller. The following topics can help you get started as an AWS Marketplace seller.

- Getting started as a seller (p. 3)
- Preparing your product (p. 29)
- Submitting your product for publication (p. 142)
- Seller reports and data feeds (p. 153)

If you're new to providing products on AWS Marketplace, the following topics can help you better understand the kinds of products available:

- AMI-based products (p. 61)
- Software as a service (SaaS)–based products (p. 113)

**Putting an agreement in place with an ISV**

Before you as a consulting partner can create a private offer for a product, the ISV must authorize you to resell their product. The ISV does this by creating an **opportunity** for you. For more information, see ISV setup of resell opportunities (p. 48). To create an opportunity, the ISV must provide:

- The product or products that they authorize you to resell.
- The price reduction that they want to offer you.
- The AWS account ID that you used to register as an AWS Marketplace seller.

After the opportunity has been created, you will be an authorized reseller for that product. Then, you can extend private offers that are marked up from the price given you by the ISV.

**Extending a private offer based on an opportunity**

For recurring discount private offers, an ISV authorizes a consulting partner to resell one or more of their products on AWS Marketplace. The discount, called the wholesale price, is an agreed-to price or percentage discount off the product’s list price. Consulting partners can use the discount with any number of buyers.

The following procedure outlines how a consulting partner can extend a private offer based on a recurring discount.

**To extend a private offer based on a recurring discount**

1. Determine what your offer price will be by marking up the wholesale price by a percentage.
3. Choose the **Offers** tab.
4. Choose the **Product for private offer**, the **Buyer account id(s)**, and then choose **Next**.
5. Choose the pricing details of the offer (including the marked-up price by a percentage), provide the End User License Agreement (EULA), and the dates for the offer. Then, choose **Review Offer**.
6. Review the offer, and then choose **Extend Offer**.

The publishing process for this offer can take up to 45 minutes to complete. After it's completed, the offer is visible on the **Manage Offers** page.

### Accepted offers

After the buyer accepts the private offer from the consulting partner, the offer and any disbursement of funds occur in the same manner:

1. AWS Marketplace invoices the buyer on their existing AWS bill per the terms of the private offer. If the private offer is extended to a linked account, the invoiced amount appears on the payer account associated with that linked account.
2. The buyer pays their AWS bill in accordance with the net payment terms that they agreed to with AWS. The private offer process enables custom terms for each transaction, but net payment terms aren't customizable.
3. After AWS receives payment from the buyer, AWS disburses payment to you and the ISV. The ISV receives the wholesale cost minus the AWS Marketplace fee. You receive your markup minus the AWS Marketplace processing fee. All fees are percentages applied to the transaction amounts listed. If you're not sure of the fee percentages and need this information for quoting purposes, contact your AWS Marketplace channel account manager. If you don't know who that is, send an email message to the AWS Marketplace channel team at <aws-mp-channel@amazon.com>, and someone on the team will respond to you within 24 hours.
4. AWS Marketplace provides electronic reports to the ISV and to you using the AWS Marketplace Management Portal. These reports have the following differences depending on the type of private offer:
   - For recurring discount private offers, the ISV sees you as the buyer and you see the subscriber as the buyer.
   - For non-recurring discount private offers, the ISV and the consulting partner see the subscriber as the buyer.

For more information about AWS Marketplace reporting, see [Seller Reporting](#).

### Private offer upgrades and renewals

For SaaS contract and SaaS contract with consumption products, you can offer upgrades and renewals by using a private offer on any active agreements. For example, you can do this to grant new entitlements, offer pricing discounts, adjust payment schedules, or change the end user license agreement (EULA) to use [standardized license terms](#) (p. 52). You can also change the number of units and payment schedule, and add a custom end date.

The difference between an offer and an agreement is whether the buyer accepted its terms:

- **An offer** is a set of terms for a buyer's use of a product. Offers can be public or private.
- **An agreement** is an offer that a buyer accepted. Agreements include purchased and free products that a seller made available via a public or private offer.

This page describes how to amend active agreements for SaaS contract and SaaS contract with consumption products.
This feature is available to all AWS Marketplace sellers, including independent software vendors (ISVs) and consulting partners. You can't amend an agreement to specify a seller of record that's different from the seller of record from the original agreement.

To use this feature, you must have permissions to use the Agreements tab in the AWS Marketplace Management Portal. For information, see Permissions for AWS Marketplace sellers (p. 207).

Supported product types

Currently the following product types support private offer renewals and upgrades:

- SaaS contracts
- SaaS contracts with consumption

Submission process for upgrades and renewals

You can create private offer upgrades and renewals from the AWS Marketplace Management Portal by using the following procedure.

To create private offer upgrades and renewals

1. Sign in to the AWS Marketplace Management Portal and choose Agreements.
2. On the Agreements page, create an upgrade or renewal private offer in one of the following ways:
   - Choose a check box next to an agreement, and then choose Create agreement-based offer.
   - Choose an agreement ID to view the agreement details. On the Agreement summary page, review the agreement's existing information and terms to verify that this is the agreement you want to amend, and then choose Create agreement-based offer.
3. On the Agreement offer details page, enter a custom offer name.
   - Tip
     Entering descriptive custom offer names can help you distinguish between your active offers on the Offers page. Custom offer names are also visible to buyers. AWS recommends that you specify a custom offer name that includes any additional identifying details, such as your own IDs and purchase order numbers. Using high-level descriptions like upgrade or renewal and custom company names are also recommended. Don't use any personally identifiable data (for example, first or last names, phone numbers, or addresses). You can enter up to 150 characters for this field.
4. Edit the information for any dates, dimensions, payment schedule, and EULA that you want to change. Then choose Next.
5. On the Review and create page, review the information. When ready, choose Create agreement-based offer.

The new private offer appears on the Manage Private Offer page in approximately 45 minutes. To view the offer, sign in to the AWS Marketplace Management Portal and choose Offers to open the Manage Private Offer page.

Similar to the process for creating a private offer, the buyer isn't notified that you created a new private offer. Instead, you provide the URL for the new private offer to the buyer. From there, the buyer has the option to accept it or to continue to operate under the original agreement:

- If the buyer accepts the private offer upgrade or renewal, the new agreement takes effect immediately and the agreement is listed on the Agreements page in the AWS Marketplace Management Portal. Any remaining scheduled payments from previous agreements are cancelled.
Buyers accept agreement-based private offers the same way they accept private offers. For more information about the buyer experience for private offers, see Private offers in the AWS Marketplace Buyer Guide.

- If the buyer doesn't accept the private offer upgrade or renewal before it expires, the original agreement remains in effect with no changes.

**Reporting for upgrades and renewals**

Upgrade and renewal private offers appear on the existing seller reports and in the reports relevant to the offer. The section called “Daily customer subscriber report” (p. 161) report and the section called “Daily business report” (p. 154) report are generated daily. The section called “Monthly billed revenue report” (p. 168) report is generated monthly.

In the Daily customer subscriber report, the Subscription intent field indicates whether the report entry is a new private offer. The Previous offer ID field indicates the ID of the offer that preceded the new offer, if one exists. For all private offers, the entry is marked private.

Currently, agreements data is not shown in data feeds.

**Standardized license terms**

As you go through the process of preparing your product (p. 29), you need to determine what to include in the end user license agreement (EULA) for your product. You can create and customize your own EULAs, or you can use the standardized license terms that AWS Marketplace offers, which help speed up transactions and simplify procurement.

AWS Marketplace offers the following options for providing standardized license terms for your product listings:

- **Standard Contract for AWS Marketplace (SCMP) (p. 52)** – These license terms are intended to meet the fundamental requirements of buyer and seller.

  This option is available to all AWS Marketplace sellers and is accessible to all buyers.

- **Enterprise Contract for AWS Marketplace (ECMP) (p. 53)** – These license terms are intended to meet the more stringent requirements of enterprise buyers.

  This option is available for all AWS Marketplace sellers to enroll in, and it's accessible only to enrolled enterprise buyers. When you enroll as a seller in the ECMP program, you can still offer SCMP to non-enterprise buyers.

**Disclaimer**

You are responsible for determining whether these documents meet your specific requirements. These documents should not be construed as legal advice for any particular facts or circumstances.

**Standard Contract for AWS Marketplace (SCMP)**

SCMP is a standardized set of license terms that govern usage and define obligations of buyers and sellers. AWS Marketplace sellers can offer SCMP as the EULA for public product listings. Buyers can search for, buy, and quickly deploy software from sellers that offer the terms of the standard contract. For private offers (p. 43), you can amend the SCMP to address custom transaction requirements as agreed upon by both parties.
Getting started with SCMP

This section describes how to review terms and offer SCMP to buyers.

To use SCMP for new and existing AWS Marketplace listings

1. Review the terms of the Standard Contract for AWS Marketplace.
3. As you create a product or edit an existing listing, choose SCMP EULA as the EULA.

For more information about creating products, see Submitting your product for publication (p. 142).

To request assistance in updating the EULA to the SCMP

1. From the lower-left corner of the AWS Marketplace Management Portal, choose Contact us.
2. Enter your email address, and then complete the rest of the form as follows:
   - For the subject of your question, choose Commercial Marketplace.
   - For the category, choose Product Listing.
   - For the subcategory, choose Standard Contract Request.
   - In the text box for providing request details, type Enable SCMP for AWS Marketplace product listings.

If you have questions about the SCMP program, contact <aws-mp-standardcontract@amazon.com>.

Enterprise Contract for AWS Marketplace (ECMP)

Like SCMP, ECMP is a standardized set of license terms that govern usage and define obligations of buyers and sellers. For private offers (p. 43), you can amend the ECMP to address custom transaction requirements as agreed upon by both parties.

The differences between ECMP and SCMP are as follows:

- ECMP is designed to address the more stringent requirements of large enterprises.
- To offer the ECMP on your listings, you must enroll in the program.
- You can offer ECMP as the EULA on public product listings and private offers only to enrolled enterprise buyers.

After enrolling in the ECMP program, you can still offer SCMP as your public EULA to address license requirements of non-enterprise customers.

Getting started with ECMP

This section describes how to review terms and enroll in the ECMP program.

To enable ECMP for AWS Marketplace listings

1. Review the terms of the Enterprise Contract for AWS Marketplace.
2. Complete the enrollment form and choose Register Now.

If you have questions about enrollment in the ECMP program, contact <aws-mp-enterprisecontract@amazon.com>.
Categories and metadata

Here are best practices and information for supplying product metadata. AWS Marketplace revises product metadata solely for quality assurance and error correction.

Naming and describing your product

The information that you provide about your product is visible to buyers. Ensure that potential buyers have enough information to make informed decisions about buying your product.

Creating the product name

Keep the following guidelines in mind as you create the product name:

- Use title case (capitalize the first letter of each important word)
- Ensure that a buyer can identify the product by the name alone
- Use the name of the brand or manufacturer
- Avoid descriptive data or hyperbole

Example product name: Smart Solution Load Balancer - Premium Edition.

Writing the product description

The product description lists the product’s features, benefits, and usage. It can also provide other relevant, specific product information. The description can be up to 350 characters long.

Keep the following guidelines in mind as you write the product description:

- Avoid unnecessary capitalization
- Avoid unnecessary punctuation marks
- Don’t include redirect information
- Check spelling and grammar
- Include only critical, useful information

Example product solution: Smart Solution automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables you to achieve even greater fault tolerance in your applications, providing the amount of load-balancing capacity you need to respond to incoming application traffic. Smart Solution detects unhealthy instances in a pool and automatically reroutes traffic to healthy instances until the unhealthy instances are restored. You can enable Smart Solution in a single AWS availability zone or across multiple availability zones to ensure more consistent application performance.

Writing the product highlights

The product information page displays up to three product highlight bullet points. Use these bullet points to briefly describe the product’s primary selling points.

Example product highlight: Projecting costs: With Smart Solution, you pay only for what you use. You’re charged for each hour or partial hour that Smart Solution is running.

Writing the release notes

Each time you update an AMI product, you must provide a description of the changes in the release notes. The release notes should contain specific information to help the user decide whether to install
Choosing categories and keywords

When you list your product, you can choose up to three software categories and corresponding subcategories for your product. This helps buyers discover your product as they browse or search for products on AWS Marketplace. Choose only categories that are relevant to your product; in most cases, only one category applies. The product load form and the Products tab both contain a complete list of categories.

Categories aren't the same as keywords. The categories and subcategories available are predefined for AWS Marketplace, and you decide which ones apply to your product by selecting them from a list during the product request process. Keywords aren't predefined, but are created during the process. You don't need to add the category as a keyword.

Writing the usage instructions

Provide usage instructions that help ensure that the buyer can successfully configure and run the software. The usage instructions you provide are shown during the AMI configuration process.

To write effective usage instructions, follow these guidelines:

• Write them with a new or moderately technical audience.
• Don't assume that the user has prior experience with or extensive knowledge of the product, computer operating systems, engineering, or IT operations.
• Take the buyer from launching to using the product, including any configuration or special steps to get the application running.

Example usage instructions:

1. Launch the product via 1-Click.
2. Use a web browser to access the application at https://<EC2_Instance_Public_DNS>/index.html.
3. Sign in using the following credentials:
   • Username: user
   • Password: the instance_id of the instance

Writing the upgrade instructions

Provide details on how buyer can upgrade from an earlier version of the product. Include information on how to preserve data and settings when creating another instance. If there is no upgrade path, edit this field to specifically mention that.

Example upgrade instructions:

1. Do ****, and then ****.
2. Check that all plugins used by your project are compatible with version *,*, by doing ***. If they aren't compatible, do ***.
3. Make a backup of your data, by doing ***.
Creating search keywords

During the product request process, you can enter up to three keywords (single words or phrases) to help buyers discover your product through site searches. The keywords field can contain a maximum of 250 characters.

The following tips can help you to create a relevant set of search keywords:

- Use relevant terms.
- Don't use the names of products published by other sellers or use other sellers' names.
- Choose keywords from your buyer's vocabulary—that is, words and phrases that buyers are likely to use when thinking about your type of product.
- Create keywords based on specific features in your product.
- Don't use the product title as a keyword. The product title is already indexed in searches.

Note: Keywords aren't the same as software categories. Keywords are more specific terms that are related to your product.

Search engine optimization

The AWS Marketplace website ranks the results of search queries using search-optimization techniques similar to those used across the industry. By understanding how AWS Marketplace ranks and returns search results, you can create product details optimized for the AWS Marketplace search engine. We recommend taking this guidance into consideration when you create your product detail pages.

Keywords

During the product creation process, you can submit up to three keywords (single words or phrases) to help customers discover your product through site searches. The keywords text box can contain up to 250 characters. Use the following tips to create search keywords:

- Use terms that are relevant so that customers can easily find your products.
- Choose keywords from your customers' vocabulary—that is, words and phrases that they're likely to use when thinking about your type of product.
- Create keywords based on specific features in your product.
- Don't include the product title in the terms that you submit. The product title is already indexed in the search.

Note: Keywords aren't the same as software categories. Keywords are more specific terms that are related to your product.

Software categories

When you list your product, you can choose up to three software categories and corresponding subcategories for your product. This helps customers discover your product as they browse or search the products on AWS Marketplace. Choose only categories that are relevant to your product. In most cases, only one category applies. Both the product load form and the Products pages contain a complete list of categories.
Note
Categories aren't the same as keywords. The available categories and subcategories are predefined for AWS Marketplace. You decide which of them apply to your product by choosing them from a list. Keywords aren't predefined, but are created during the process.

Highlights section

The product details page displays up to three product highlights as bullet points. Customers can search for products by highlights, so include highlights when you create a product. A highlight should describe the product's primary selling points in brief and informative language. For example: "Projecting costs: With AnyCompany's Smart Solution, you pay only for what you use. You're charged for each hour or partial hour that Smart Solution is running."

Short description

The product description lists the product's features, benefits, and usage instructions, along with other relevant and specific product information. Keep the following guidelines in mind as you create the product description:

- Avoid unnecessary capitalization and punctuation marks
- Don't include redirect information
- Check spelling and grammar
- Include only critical and useful information

Example

AnyCompany's Smart Solution automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables you to improve fault tolerance in your applications by seamlessly providing the load balancing capacity that you need to respond to incoming application traffic. Smart Solution detects unhealthy instances in a pool and automatically reroutes traffic to healthy instances until the unhealthy instances have been restored. Customers can enable Smart Solution in a single AWS Availability Zone or across multiple Availability Zones to enable more consistent application performance.

The AWS Marketplace Managed Catalog Operations team helps redirect queries with similar-sounding words or words with similar meanings: for example, when customers search for automobile when you expect them to search for car.
AWS Marketplace for Desktop Applications (AMDA)

AWS Marketplace for Desktop Applications (AMDA) is a catalog of virtualized desktop applications that run on Amazon WorkSpaces. AMDA makes it easy to find and subscribe to free and paid applications across 11 software categories. Applications run in virtualized containers as if they were natively installed and buyers are charged on a per-user, per-month basis.

Buyers use the Amazon WorkSpaces Application Manager (WAM) console to deploy desktop applications to their WorkSpaces. The applications are delivered to each WorkSpace through the WAM client application.

The virtualization technology enables fast delivery of programs, often without a reboot, so that users can quickly launch and use their subscribed applications. Users are charged only for those applications they have been assigned, and charges accrue monthly from when they are first launched until the assignment is revoked. Additional information:

- Amazon WorkSpaces product pages
- Amazon WorkSpaces testimonials
- AMDA help pages and frequently asked questions
- AWS Marketplace for Desktop Applications catalog

Starting the onboarding process

Under the terms of our AWS Marketplace for Desktop Applications Publisher Addendum (the “AMDA Addendum”), Amazon Web Services, Inc. is the seller of record for applications you choose to make available through the AMDA channel. As the seller of record, AWS will need to know the price you will charge AWS for the products you plan to have on AWS Marketplace. Pricing should be on a per-month basis, per user. AWS will help you to determine the final price to buyers.

In order to have your product published in non-US Regions, AWS will also need you to provide certain export classification information, including the applicable Export Control Classification Number (ECCN).

During and after the initial testing of your product(s), a member of the AMDA Business Development team be available to answer any question you might have. You will then receive an email from <aws-mp-amda-contract@amazon.com> with the AMDA Addendum for you to fill out, sign, and return for counter signature. Note that the AMDA Addendum is an addendum to the Terms and Conditions for AWS Marketplace Sellers, so you will need to establish an AWS Marketplace seller account and click through these terms prior to beginning the onboarding process to have your applications made available on AMDA.

Product submission and packaging

Virtualization and packaging are handled by the AWS Marketplace Managed Catalog Operations (MCO) team. AMDA vendors provide the software installer, installation instructions, and product metadata. MCO will work with you to complete the packaging and complete the process for AMDA. Currently, all AMDA software must be packaged by using an MCO administrative account with permissions to the Amazon S3 bucket that will store the package. AWS is unable to accept shared packages. Review the following
guidelines before you submit your product. MCO will start processing your packaging request upon receipt of these items:

1. Software installer and license key:
   a. Amazon S3 bucket or external URL for the hosted Installer file (.msi, .exe, etc.)
   b. Server license key that is compatible with Windows Server 2008 R2

2. Installation instructions:
   a. Known issues for Windows Server 2008 R2
   b. Silent install command line arguments
   c. Licensing mechanism notes:
      i. Where is the license stored?
      ii. How is the license verified?
      iii. Which actions trigger a license check?
   d. Auto-update
      i. If enabled, describe how to disable this function
   e. Services or Registry requirements:
      i. List each required service or registry key, with a brief description of its purpose

3. Test servers, data files, and additional external elements
   a. If required for installation, provide a test environment for external components (for example, SQL Server)
   b. If your program processes data files, include test files so we can ensure performance and functionality

4. List all program dependencies, for example:
   a. C++ redistributables
   b. Java, QuickTime, etc.
   c. GPU/hardware requirements

5. Program technical contacts
   a. Who is the point of contact for technical questions or issues encountered during testing and packaging?

Application packaging types

AMDA packaging can be completed in two ways: virtualized installation or silent installation.

Virtualized installation relies on AMDA packaging tools to monitor all file changes during the installation process. AWS will point to the installer executable and click **Install**, which will monitor all file changes. AWS then makes custom changes to the registry, services, and file structure to ensure program stability and performance.

Some advanced programs require a silent installation mechanism. In this case, AMDA will virtualize only the installer files so that the software is physically installed only when the application is first launched on the user’s WorkSpace. Additional steps are required to script the removal of silent installation programs.

Building the AMDA package

The packaging process relies on creating a diff of the target installation machine, which is a Windows Server 2008 R2 virtual machine (VM). The packaging tool monitors the VM during the installation process, creates a manifest of the changed files, and rolls this into a package to be ingested.
After capturing the changes programmatically, an AWS technician will inspect the files, services, and registry entries to ensure all changes were accurately captured. During this process, the technician will remove all uninstall and auto-update references to ensure the application stays within the confines of the virtualized package.

Programs that rely on specific Windows services (background-running Windows services, .dll requirements, etc.) might require additional testing and packaging. By default, all program properties are virtualized to run on demand. Some services might require elevation to ensure they are available to the program at runtime.

License keys will be captured during the packaging process to help ensure a seamless, one-click experience for end users. If your program requires the license keys on first launch, include detailed notes about how to manually add the license to the applications files.

Application metadata

Enter the application metadata into the AMDAPasswordDataLoad.xlsx load form and include it with your application submission. The current data load form is always available at https://s3.amazonaws.com/aws-mp-vendor-guide/AMDAProductDataLoad.xlsx

- Title – This is the title of the product.
- Full Description – This appears on the product detail page.
- Short Description – This appears on the search results page.
- End User License Agreement – This is the EULA that applies to the buyer’s use of the product.
- Image – This is the product image or logo that appears on the product detail page, in search results, and elsewhere on the AMDA website. Provide a URL to a square-formatted image logo
- Categories – This is the software category for the product. See the AMDA home page to view the available categories.
- Software By – This is the software developer that is displayed on the product page, which is usually your company name.
- Vendor URL – This is the link to your website or a specific page that displays more information on the product.
- Support text/email/URL (Only one field is required, but multiple contact points are encouraged)

Ingestion and new version updates

Ingestion of each AMDA product is handled by the AWS Marketplace MCO team. The current pipeline supports releases on Thursday. AWS will lock on metadata and final packaging on Tuesday at noon PST. Requests after Tuesday noon PST will be eligible on the following week’s publishing day. New version updates are made on the same schedule.

If no metadata updates are requested, only the installer and associated files are required.

If you are updating metadata, send an updated product data load form to the AWS Marketplace Seller Operations team.
AMI-based products

Amazon Machine Images (AMIs) provide the information required to launch an Amazon EC2 instance.

Each product in AWS Marketplace is assigned a unique product ID. This product ID is used to identify your product in the AWS Marketplace catalog, in customer billing, and in seller reports. A unique product code is assigned to all AMIs submitted to AWS Marketplace. Product codes are not product IDs. Sellers can obtain the product code while they develop their software so it can be used for extra security, such as validating the product code at product start. You cannot make API calls to an AMI’s product code until the product has been published into a limited state for testing.

Product codes are propagated automatically as customers work with the software. For example, a customer subscribes and launches an AMI, configures it, and produces a new AMI. The new AMI still contains the original product code, so correct billing and permissions remain in place. For more information, see Instance Metadata and User Data in the Amazon EC2 User Guide for Linux Instances.

Multiple versions

You can provide multiple versions of a product as separate AMIs to buyers as part of their purchase. The seller can make available any number of versions for a product. After a buyer has access to an AMI, they always have launch permissions on the AMI, regardless of the visibility or status of that version.

For example, the product Data Cleaner might have versions 1.0.0, 1.2.5, and 2.0.1, all of which can be available to buyers. If you request the removal of version 1.0.0, no new customers can buy that version, but existing customers can access it.

AMI file upload

Self-service AMI scanning is available in the AWS Marketplace Management Portal. With this feature, you can initiate scans of your AMIs and receive scanning results quickly—typically in less than an hour—with clear feedback in a single location. For more information, see AMI Self-Service Scanning.

To upload a new product load form, go to File Upload in the AWS Marketplace Management Portal. From there, you can download the most recent product load template. We strongly recommend checking that the form is the most recent because it’s consistently updated with more instance types and regions as they become available. Using AMI Self-Service Scanning significantly increases the ease of loading the page.

Removing products from AWS Marketplace

After your product is published, you can remove (also referred to as sunset) the product from AWS Marketplace. To remove a product, identify the product and submit a request to remove it, along with a reason for removal and a contact email address for you. You can also provide a replacement product ID if you’re replacing the current product with a new one. After you request product removal, new customers will no longer be able to subscribe. You’re required to support any existing customers for a minimum of 90 days. We process requests for product removal from AWS Marketplace with the following conditions:

- The product is removed from AWS Marketplace search, browse, and other discovery tools. Any Subscribe button or functionality is disabled, and messaging on the page clearly indicates the product is no longer available. Note that the product detail page is still accessible using the URL and may be indexed in public search engines.
• A reason for removal must be specified (for example, end of support, end of product updates, or replacement product). For the requirements for continuing support for removed products, see Terms and Conditions for AWS Marketplace Sellers.
• Current buyers are messaged by AWS Marketplace informing of the product removal, reasons for the removal, and provide seller contact information.
• Current buyers do retain access to the software until they cancel their subscription. They aren’t impacted in any way by the product removal.

To remove a product created using the AWS Marketplace Management Portal

2. Choose the Products tab, and then choose Server.
3. On your product page under Current server products, locate the product that you want to remove. From the Actions column on the Select action menu, choose Remove product.
4. On the Remove Product page, for Request Reason, type the reason that you’re requesting the product’s removal.
5. For Contact Email, type the email that AWS can use to contact you with any questions.
   Note
   You can also provide a replacement product ID, but that field isn’t required.
6. Review the information for accuracy, and then choose Submit Sunset Request.

A What’s next informational page displays after you submit the product removal request. The AWS Marketplace Seller Operations team reviews and processes your request. Check the status of your submission by viewing Requests.

After your product is removed, the product appears in your Request History list and in the Current Products list. In Current Products, the only action that you can perform is downloading the spreadsheet for the product. You can’t edit or submit another sunset request.

For products not created using the Products tab, edit and upload the product load form for the product. Links to upload updated product load forms are on the Assets tab on the AWS Marketplace Management Portal landing page.

If you have questions about product removals, contact the AWS Marketplace Seller Operations team.

Best practices for building AMIs

All Amazon Machine Images (AMIs) built and submitted to AWS Marketplace must adhere to all AWS Marketplace product policies. To share your AMI and verify that it meets all AWS Marketplace requirements, use the self-service AMI scanning tool. This page provides some best practices and references to help you build AMIs.

Rights

You are responsible for securing resell rights for non-free Linux distributions, with the exception of AWS-provided Amazon Linux, RHEL, SUSE, and Windows AMIs.

Building an AMI

Use the guidelines for building AMIs:
Securing an AMI

- Ensure that your AMI meets all AWS Marketplace policies, including disabling root login.
- Create your AMI in the US East (N. Virginia) Region.
- Create products from existing, well-maintained AMIs backed by Amazon Elastic Block Store (Amazon EBS) with a clearly defined life cycle provided by trusted, reputable sources such as AWS Marketplace.
- Build AMIs using the most up-to-date operating systems, packages, and software.
- Ensure that all AMIs must start with a public AMI that uses hardware virtual machine (HVM) virtualization and 64-bit architecture.
- Develop a repeatable process for building, updating, and republishing AMIs.
- Use a consistent operating system (OS) user name across all versions and products. We recommend `ec2-user`.
- Configure a running instance from your final AMI to the end-user experience you want and test all installation methods, features, and performance before submission to AWS Marketplace.
- Check port settings.
  - For Linux-based AMIs, ensure that a valid SSH port is open. The default is 22.
  - For Windows-based AMIs, ensure that an RDP port is open. The default is 3389. Also, the WinRM port (5985 by default) must be open to 10.0.0.0/16.

Resources:

- Creating Your Own AMI in the *Amazon EC2 User Guide for Linux Instances*
- Creating a Custom Windows AMI in the *Amazon EC2 User Guide for Windows Instances*
- How do I create an Amazon Machine Image (AMI) from an EBS-backed Windows instance?
- Amazon Linux AMI
- Amazon EC2 Instance Types and Instance Types

### Securing an AMI

The following guidelines are recommended for creating secure AMIs:

- Architect your AMI to deploy as a minimum installation to reduce the attack surface. Disable or remove unnecessary services and programs.
- Whenever possible, use end-to-end encryption for network traffic. For example, use Secure Sockets Layer (SSL) to secure HTTP sessions between you and your buyers. Ensure that your service uses only valid and up-to-date certificates.
- Use security groups to control inbound traffic access to your instance. Ensure that your security groups are configured to allow access only to the minimum set of ports required to provide necessary functionality for your services. Allow administrative access only to the minimum set of ports and source IP address ranges necessary.
- Consider performing a penetration test against your AWS computing environment at regular intervals; or, consider employing a third party to conduct such tests on your behalf. For more information, including a penetration-testing request form, see AWS Penetration Testing.
- Be aware of the top 10 vulnerabilities for web applications, and build your applications accordingly. To learn more, visit Open Web Application Security Project (OWASP) - Top 10 Web Application Security Risks. When new internet vulnerabilities are discovered, promptly update any web applications that ship in your AMI. Examples of resources that include this information are SecurityFocus and the NIST National Vulnerability Database.

Resources:
The AWS Marketplace Metering Service is a pricing and metering feature that sellers can use to directly charge for their software by one of four usage categories: users, data, bandwidth, or hosts. You can use the Metering Service with AMI-based, container-based, and SaaS-based products. For more information, see the [AWS Marketplace Metering Service API Reference](https://docs.aws.amazon.com/marketplace/metering-service/api).

All AMI-based software that uses the Metering Service must meet the following requirements:

- Your software must be launched from AWS Marketplace through an Amazon Machine Image (AMI).
- If you have an existing product in AWS Marketplace, you must submit a new AMI and create a new product to enable this feature.
- All software must be provisioned with an AWS Identity and Access Management (IAM) role. The end customer must add an IAM role to the Amazon Elastic Compute Cloud (Amazon EC2) instance the user is provisioning with the software. Currently, the use of an IAM role is optional when you deploy software through AWS Marketplace. It is required when you deploy AWS Marketplace Metering Service software.
- Your software must be able to determine consumption in some way.

Products that use the Metering Service must charge customers along a single usage category, but you can define up to 24 dimensions of a single category. Depending on the category, software can be priced by provisioned resources, concurrent resources, or accumulated resource consumption. All charges are still incurred hourly by the customer. All usage is calculated and billed monthly using the same mechanism as existing AWS Marketplace software.

The AWS Marketplace Metering Service enables several new scenarios. For example, if your software monitors hosts, you can charge for each host monitored. You can have different prices based on the host size, and charge for the number of concurrent hosts monitored each hour. Similarly, if your software allows many users across an organization to sign in, you can charge by the number of users. Each hour, the customer is charged for the total number of provisioned users.

**Metering service concepts**

The AWS Marketplace Metering Service enables software sellers to modify their software to send metering records to an endpoint to capture usage. Sellers can select a usage category and define up to 24 dimensions of that one category. These dimensions are metered once per hour, aggregated, and charged against a price plan defined by the seller. As a seller, the first thing you need to do is determine which dimension you want to use. After the AMI is published, you will not be able to change it. Important service concepts include the following:

- **Usage Category** – Any software product priced through the use of the Metering Service will select one usage category along which to charge customers, including but not limited to:
• Users – A defined set of permissions associated with a single identifier. This category is appropriate for software in which a customer’s users connect to the software directly (for example, for customer-relationship management or business intelligence reporting).

• Hosts – Any server, node, instance, endpoint, or other part of a computing system. This category is appropriate for software that monitors or scans many customer-owned instances (for example, performance or security monitoring).

• Data – Storage or information, measured in MB, GB, or TB. This category is appropriate for software that manages stored data or processes data in batches.

• Unit – Unit of measurement; see the examples described next.

• **Usage Unit** – A software product will select a specific usage unit corresponding to the selected usage category. This is usually more specific and describes the unit your software will charge on. Examples include:
  - NodesHrs (corresponding to the Hosts category)
  - UserHrs (corresponding to the User category)
  - GBStored (corresponding to the Data category)

• **Consumption** – Any software product priced through the use of the Metering Service will charge for consumption in one of three ways:
  - Provisioned – The software allows customers to configure a specific amount of resources for use (for example, number of users or a fixed amount of bandwidth). Each hour, customers pay for what they have provisioned.
  - Concurrent – The software allows any number of distinct hosts or users to connect to the software. Each hour, customers pay based on the number of hosts or users who accessed the software.
  - Accumulated – The software allows customers to use any amount of data, either processed or stored. Each hour, customers pay for the aggregated amount.

• **Pricing** – Any software product priced through the use of the Metering Service must specify either a single price or define up to 24 dimensions, each with their own price.
  - Single dimension – This is the simplest pricing option. Customers pay a single price per resource unit per hour, regardless of size or volume (for example, $0.014 per user per hour, or $0.070 per host per hour).
  - Multiple dimensions – This pricing option is appropriate when the selected usage category varies along multiple axes. For example, for host monitoring, a different price could be set depending on the size of the host. Or, for user-based pricing, a different price could be set based on the type of user (for example, admin, power user, and read-only user).

• **Metering** – All usage is recorded as a metering event, once each hour. Your software must be configured to send the appropriate dimension and usage amount to the AWS Marketplace Metering Service.

• Allocations – Optionally, you may distribute the usage into allocations by properties that you track. These allocations are represented as tags to the buyer. The tags allow the buyer to view their costs split into usage by tag. For example, if you charge by the user, and users have a “Department” property, you could create usage allocations with tags that have a key of "Department", and one allocation per value. This does not change the price, dimensions, or the total usage that you report, but allows your customer to view their costs by categories appropriate to your product.

---

**Pricing your software**

When pricing your software with the AWS Marketplace Metering Service, you must first decide on a usage category and how it will be consumed. At this time, the service supports six distinct pricing scenarios. You must select only one of these for your product:

• Provisioned user (per hour)
• Concurrent user (per hour)
Next, you must decide how to price the selected usage category:

- Single price
- Multiple dimensions (up to 24)

the section called “Adding your product to AWS Marketplace ” (p. 66) describes how to provide a customer-friendly description of your dimension and pricing.

**Example: Provisioned bandwidth with nonlinear pricing**

Imagine you offer network appliance software. You choose to bill by provisioned bandwidth. For your usage category, select bandwidth. In addition to charging by bandwidth, you want to charge a different price as buyers scale up. You can define multiple dimensions within the bandwidth category. You can define a distinct price for 25 Mbps, 100 Mbps, and 1 Gbps.

**Example: Concurrent hosts with multiple dimensions**

Imagine you offer software that monitors other Amazon EC2 instances. You choose to bill by the number of hosts that are being monitored. For your usage category, select host. In addition to charging by host, you want to charge for the extra value for monitoring larger hosts. You can use multiple dimensions within the host category. You can define a distinct price for micro, small, medium, large, x-large, 2XL, 4XL, and 8XL instances. Your software is responsible for mapping each particular host to one of your defined dimensions. Your software is responsible for sending a separate metering record for each dimension of your usage category if applicable.

**Adding your product to AWS Marketplace**

To take advantage of the Metering Service, you must create a new product for AWS Marketplace to list. If your product is already on the AWS Marketplace, you will need to decide whether the new AWS Marketplace Metering Service product will be made available in addition to your current product, or if it will replace your current product as the only version available to new users. If you choose replacement, the existing product will be removed from the AWS Marketplace so that it is no longer available for new buyers. Existing customers will continue to have access to their old product and instances, but they can migrate to the new product at their convenience. The new product must meter usage to the AWS Marketplace Metering Service, as described in Modifying your software to use the Metering Service (p. 67).

After you have your AMI, follow the standard process to share and scan your AMI using the self-service tool. In addition to using the template available on the management portal, fill out the product load form and upload it to start the ingestion process.

Use the following definitions to complete the fields of the product load form for the AWS Marketplace Metering Service. On the product load form, these fields are labeled as Flexible Consumption Pricing (FCP) to differentiate them from hourly and monthly priced products.

- **Title:** If you already have a product on AWS Marketplace and you are adding the same product with the AWS Marketplace Metering Service, include the FCP category/dimension in parentheses to differentiate the two (for example, “PRODUCT TITLE (Data)”).
- **Pricing Model:** From the dropdown list, choose Usage.
Modifying your software to use the Metering Service

You will need to modify your software to record customer usage, send hourly usage reports to the Metering Service, and handle new failure modes. The software operates independently of pricing, but the software will need to know about the usage category, how it is consumed, and any dimensions.

Measuring consumption

Your software must determine how much of the selected usage category and which dimensions the customer has consumed. This value will be sent, once each hour, to the Metering Service. In all cases, it

• **FCP Category**: The category in which customers are charged for paid products with a **Usage** pricing component. From the dropdown list, choose **Users**, **Hosts**, **Data**, or **Bandwidth**.

• **FCP Unit**: The unit of measurement on which customers will be charged for paid products with a **Usage** pricing component. Options will appear in the dropdown list based on the FCP category you selected. The following table lists the valid units for each category.

<table>
<thead>
<tr>
<th>Category</th>
<th>Valid units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>UserHrs</td>
</tr>
<tr>
<td>Hosts</td>
<td>HostHrs</td>
</tr>
<tr>
<td>Data</td>
<td>MB, GB, TB</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>Mbps, Gbps</td>
</tr>
</tbody>
</table>

• **FCP Dimension Name**: The name used when sending metering records by calling the **MeterUsage** operation. It is visible in billing reports, but because it is not external-facing, the name does not need to be user-friendly. The name can be no more than 15 characters and can only include alphanumeric and underscore characters. After you set the name, you cannot change it. Changing the name requires a new AMI.

• **FCP Dimension Description**: The customer-facing statement that describes the dimension for the product. The description (for example, Administrators per hour, Per Mbps bandwidth provisioned) can be no more than 70 characters and should be user-friendly. After the product is published, you cannot change this description.

• **FCP Rate**: The software charge per unit for this product. This field supports 3 decimal places.

**Notes:**

• You do not need to fill out hourly and annual pricing fields.
• Free trial and annual pricing are not compatible.
• Currently, products that use multiple AMIs and the Clusters and AWS Resources feature cannot use the AWS Marketplace Metering Service.
• Price, instance type, or Region change will follow the regular process as other AWS Marketplace products.
• Products with the AWS Marketplace Metering Service cannot be converted to other pricing models such as hourly, monthly, or Bring Your Own License (BYOL).
• AWS Marketplace recommends adding IAM policy information in your usage instructions or document.

If you have questions, contact the **AWS Marketplace Seller Operations** team.
is assumed that your software has the ability to measure, record, and read consumption of resources for
the purpose of sending it on an hourly basis to the Metering Service.

For provisioned consumption, this will typically be read from the software configuration as a sampled
value, but might also be a maximum configured value, recorded each hour. For concurrent consumption,
this might be either a periodic sample or a maximum value recorded each hour. For accumulated
consumption, this will be a value that is accumulated each hour.

For pricing on multiple dimensions, multiple values must be measured and sent to the Metering Service,
one per dimension. This requires your software to be programmed or configured with the known set of
dimensions when you provide the AMI. The set of dimensions cannot change after a product is created.

For each pricing scenario, the following table describes recommended ways for measuring consumption
each hour.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>How to measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provisioned user</td>
<td>Current number of provisioned users (sampled).</td>
</tr>
<tr>
<td></td>
<td>-OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum number of provisioned users (seen that hour).</td>
</tr>
<tr>
<td>Concurrent user</td>
<td>Current number of concurrent users (sampled).</td>
</tr>
<tr>
<td></td>
<td>-OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum number of concurrent users (seen that hour).</td>
</tr>
<tr>
<td></td>
<td>-OR-</td>
</tr>
<tr>
<td></td>
<td>Total number of distinct users (seen that hour).</td>
</tr>
<tr>
<td>Provisioned host</td>
<td>Current number of provisioned hosts (sampled).</td>
</tr>
<tr>
<td></td>
<td>-OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum number of provisioned hosts (seen that hour).</td>
</tr>
<tr>
<td>Concurrent host</td>
<td>Current number of concurrent hosts (sampled).</td>
</tr>
<tr>
<td></td>
<td>-OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum number of concurrent hosts (seen that hour).</td>
</tr>
<tr>
<td></td>
<td>-OR-</td>
</tr>
<tr>
<td></td>
<td>Total number of distinct hosts (seen that hour).</td>
</tr>
<tr>
<td>Provisioned bandwidth</td>
<td>Current provisioned bandwidth setting (sampled).</td>
</tr>
<tr>
<td></td>
<td>-OR-</td>
</tr>
<tr>
<td></td>
<td>Maximum provisioned bandwidth (seen that hour).</td>
</tr>
<tr>
<td>Accumulated data</td>
<td>Current GB of data stored (sampled).</td>
</tr>
</tbody>
</table>
### Call AWS Marketplace Metering Service

Your software must call the Metering Service hourly and record the consumption value for that hour.

When your software starts, it should record the minute-of-the-hour at which it started. This will be referred to as the *start-minute*. Every hour on the start-minute, your software must retrieve the consumption value for that hour and call the Metering Service. For information about how to obtain this value, see the section called “Measuring consumption” (p. 67) section.

To wake up each hour at the start-minute, your software must use one of the following approaches:

- A thread within your software.
- A daemon process that starts up with the instance or software.
- A cron job that is configured during application startup.

**Note**

Your software must call the AWS Marketplace Metering Service using the IAM role configured on the customer’s instance and specify the consumption dimension and amount.

Your software can use the AWS SDK to call the AWS Marketplace Metering Service, similar to the following example implementation:

1. Use the instance profile to create a service client. This requires the role configured for the EC2 instance. The role credentials are refreshed by the SDK automatically.
2. Each hour, read your software configuration and state to determine consumption values for that hour. This might include collecting a value-per-dimension.
3. Call the `meterUsage` method on the SDK client with the following parameters (call additionally for each dimension that has usage):
   - `timestamp`: Timestamp of the hour being recorded (in UTC).
   - `productCode`: Product code assigned to the software.
   - `dimension`: Dimension (or dimensions) assigned to the software.
   - `quantity`: Consumption value for the hour.
   - `allocations`: (Optional) You may provide allocations for the usage across properties that you track. These allocations must add up to the total consumption in the record. To the buyer, these display as potential cost allocation tags in their billing tools (such as the AWS Billing and Cost Management console). The buyer must activate the tags in their account in order to track their cost using these tags.

In addition, your software must call an in-Region AWS Marketplace Metering Service endpoint. Your product must have a correct regional endpoint set up, so `us-east-1` sends records to a `us-east-1`
When you send metering records to the service, you must connect to the AWS Marketplace Metering Service in your Region. Use the `getCurrentRegion()` helper method to determine the region in which the EC2 instance is running, and then pass this Region information to the `MeteringServiceClient` constructor. If you do not specify an AWS Region in the SDK constructor, the default `us-east-1` Region is used. If your application attempts to make cross-Region calls to the service, the calls are rejected. For more information, see Determining an Application's Current Region and `getCurrentRegion()`.

**Failure handling**

Your product must send metering records to the service, a public internet endpoint, so that usage can be captured and billed. Because it is possible for a customer to modify network settings in a way that prevents your metering records from being delivered, your product should account for this by choosing a failure mode.

Typically, software can fail open (provide a warning message but maintain full functionality) or fail closed (disable all functionality in the application until a connection has been reestablished). You can choose to fail open, closed, or something specific to your application. We strongly recommend that you refrain from failing closed after less than two hours of metering failures.

As an example of failing partially open, you could continue to allow access to the software but not allow the buyer to modify the software settings. Or, a buyer could still access the software, but would not be able to create additional users. Your software is responsible for defining and enforcing this failure mode. Your software's failure mode must be included when your AMI is submitted, and it cannot be changed later.

**Limitations**

Keep these limitations in mind when designing and submitting your Metering Service-enabled software:

- **IAM role and internet gateway requirements for your customers** – Your customers must have an internet gateway and must launch your software with an IAM role with specific permissions. For more information, see the section called "AWS Marketplace metering and entitlement API permissions" (p. 211). Your software cannot connect to the Metering Service if these two conditions are not met.

- **Inability to add or change new usage category or dimensions to existing Metering Service product** – When customers subscribe to your software product, they are agreeing to terms and conditions. Changing the dimensions in products with the Metering Service requires a new product and a new subscription.

- **Lack of free trial and annual subscriptions** – Metering Service products do not support free trials and annual subscriptions at launch.

- **Multi-instance or cluster-based deployment considerations** – Some software is deployed as part of a multi-instance deployment. When you design your software, consider how and where consumption is measured and where metering records are emitted.

### AMI-based delivery using AWS CloudFormation

AWS Marketplace sellers can list AMI-based products that are delivered to AWS Marketplace buyers by using AWS CloudFormation templates. You can use the templates to define a cluster or distributed architecture for the products or to select different AMI combinations or product configurations. The
AWS CloudFormation templates can be configured to deliver a single Amazon Machine Image (AMI) or multiple AMIs along with associated config files and Lambda functions. Buyers can browse the selection of solutions on AWS Marketplace, buy with one click, and deploy by using AWS CloudFormation templates that you provide.

Multi-AMI solutions can contain up to 20 AMIs and up to 20 AWS CloudFormation templates. Each AWS CloudFormation template can reference any combination or subset of the AMIs contained in the solution. The buyer purchases a single solution that entitles them to all of the AMIs in that product. When the product has multiple AMIs, each AMI has its own unique product code and can be priced and metered separately. However, individual components of a solution aren't discoverable or procurable outside the context of the product.

If you have existing single-AMI products, you can't migrate or combine them into a new multi-AMI listing. However, your new solution can feature the same software or copies of AMIs used by existing products. Each listing created on AWS Marketplace is a listing with new product codes.

You can also include Lambda functions in a Serverless Application with your AMI so that buyers can deploy them through CloudFormation. For instructions on how to include Lambda functions and serverless applications with your AMI, see Adding serverless application components (p. 74).

Building your product listing

To submit your product, you need to prepare and validate your AMI(s), create your AWS CloudFormation template(s), create a topology diagram, complete the product load form, and submit the materials to AWS Marketplace. We recommend that you start by creating and validating your AMI(s) and then complete and validate the AWS CloudFormation template(s). After you complete those steps, you should create a topology diagram and estimate the software and infrastructure price. AWS Marketplace validates your submission and works with you to make your product public. Use the AWS Pricing Calculator to help estimate the infrastructure cost for your template. Provide AWS Marketplace with a link to your saved calculator configuration. The following are limitations of multi-AMI solution products:

- Updating existing AWS Marketplace products from a standalone product to a multi-AMI product isn't supported. To make a product available in a multi-AMI product, copy the AMI and submit it as a component to a new multi-AMI product. The resulting AMI has a unique product code that's different from the previous product's code.
- Multi-AMI solutions aren't visible on the AWS Marketplace tab of the Launch Instance page in the Amazon Elastic Compute Cloud (Amazon EC2) console.
- An AWS CloudFormation template must not launch AMIs outside of those listed in the multi-AMI solution.
- AWS CloudFormation templates must be submitted in the form of a public URL. All nested template URLs contained in the template must also be publicly accessible.

Preparing your AWS CloudFormation template

To build your AWS CloudFormation templates, you must meet the template prerequisites and provide the required input and security parameters. When submitting your AWS CloudFormation template, use the guidelines in the following sections.

Template prerequisites

- Verify that the template is launched successfully through the AWS CloudFormation console in all Regions enabled for your product. You can use this tool to test your templates: https://github.com/aws-quickstart/taskcat.
- If you are creating a single-AMI product, the template must contain only one AMI.
• AMIs must be in a mapping table for each region. The AWS Marketplace team updates the AMI IDs after they're cloned.
• Build templates so that they do not depend on the use in a particular availability zone (AZ). Not all customers have access to all AZs, and AZs are mapped differently for different accounts.
• You can include dependencies such as Lambda functions, config files, and scripts with your AMI. For more information, see Create a serverless application (p. 75).
• If you’re building a clustered solution using an Auto Scaling group, we recommend that you account for a scaling event. The new node should join the running cluster automatically.
• Even for single-node products, we recommend using an Auto Scaling group.
• If your solution involves a cluster of multiple instances, consider using placement groups if you want low network latency, high network throughput, or both among the instances.
• If your solution involves Docker containers, you must incorporate the Docker images into the AMI.
• For ease of review by the AWS Marketplace team and transparency to the customer, we recommend that you add comments in your UserData section.

Template input parameters

• Input parameters to the template must not include the AWS Marketplace customer’s AWS credentials (such as passwords, public keys, private keys, or certificates) or personal information such as email address.
• Do not set defaults for parameters such as remote access, CIDR/IP, or passwords for databases. The customer must provide these as input parameters.
• For sensitive inputs such as passwords, choose the No Echo property and enable stronger regular expression. For other inputs, set the most common inputs along with appropriate helper text.
• Use AWS CloudFormation parameter types for inputs where available.
• Use AWS::CloudFormation::Interface to group and sort input parameters.

Network and security parameters

• Ensure that the default SSH port (22) or RDP port (3389) isn't open to 0.0.0.0.
• Instead of using the default virtual private cloud (VPC), we recommend that you build a VPC with appropriate access control lists (ACLs) and security groups. Only AWS accounts created before December 4, 2013, support EC2-Classic.
• Access to the customer’s AWS environment should be enabled using an IAM role to call AssumeRole from the AWS Security Token Service.
• Set IAM roles and policies to grant the least privilege and enable write access only when absolutely necessary. For example, if your application needs only S3:GET, PUT, and DELETE operations, specify those actions only. We don't recommend the use of S3:* in this case.

After your template is received, AWS Marketplace validates the product configuration and information and provides feedback for any required revisions.

Getting the cost estimate for your template infrastructure

The infrastructure cost estimate for each template displayed to customers is based on an estimate that you provide by using the AWS Pricing Calculator. The estimation should include the list of services to be deployed as part of the template, along with the default values for a typical deployment.
After you calculate the template's estimated monthly cost, provide AWS Marketplace with the **Save and Share** link for the US East (N. Virginia) Region. This is part of the submission process.

**Topology diagram**

You must provide a topology diagram for each template. The diagram must use the **AWS product icons** for each AWS service deployed through the AWS CloudFormation template, and it must include metadata for the services. The diagram must be 1100 x 700 pixels in size. Make sure that your diagram meets this sizing requirement to avoid cropping or stretching, as shown in the following image.

---

**Meeting the submission requirements**

To submit products delivered by using AWS CloudFormation templates, you must provide the following resources:

- AWS CloudFormation template or templates
  - A single-AMI product can have one to three AWS CloudFormation templates
  - A multi-AMI product can have up to 20 AWS CloudFormation templates
- The estimated infrastructure price for the default configuration of each template
- A topology diagram and topology metadata
- Completed product form (available from the AWS Marketplace Management Portal)
  - For single-AMI products, use the **Commercial Product form**
  - For multi-AMI products, use the **Multi-AMI Product form** form

The product forms include example submissions for your reference.
For each product, most of the required product data and metadata are the same as for traditional single-AMI products. Therefore, each AMI that is delivered by using an AWS CloudFormation template must continue to meet the standards and requirements described for AWS Marketplace.

For each AWS CloudFormation template, you must also provide the following information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Title of the topology. This appears on the detail and fulfillment pages and the pop-up that shows the topology details.</td>
<td>50 characters</td>
</tr>
<tr>
<td>Short description</td>
<td>This appears on the detail and fulfillment pages.</td>
<td>200 characters</td>
</tr>
<tr>
<td>Long description</td>
<td>This appears in the topology details pop-up.</td>
<td>2000 characters</td>
</tr>
</tbody>
</table>

For multi-AMI products, the following fields are required:

- Solution title
- Solution short description
- Solution long description
- For AWS CloudFormation templates (up to 20 per solution)
  - Deployment title (per template)
  - Short description (per template)
  - Long description (per template)
  - Architecture diagram (per template)
  - Infrastructure pricing estimate (per template)
  - List of products/components contained in this AWS CloudFormation template
  - List of regions supported by this AWS CloudFormation template

**Submitting your product request**

Use the AWS Marketplace Management Portal to submit your product. On the Assets tab, choose File Upload. Upload any files you want to submit and enter a brief description. Allow three to five weeks for request processing, including:

- Review of the AWS CloudFormation template, AMI, and metadata for the AMI and AWS CloudFormation template
- Publication of your AWS CloudFormation template to AWS Marketplace products

**Adding serverless application components**

You can create a product that includes one or more Amazon Machine Images (AMIs), delivered using one or more AWS CloudFormation templates, with serverless components incorporated into the product. For example, you can create a product with one AMI configured as a controller server and another AMI configured as a worker server, delivered as a AWS CloudFormation stack. The AWS CloudFormation
template used to create the stack can include the definition to set up an AWS Lambda function that is triggered by an event in one of the servers.

When you use this approach to design your product, you can simplify the architecture and make it easier for your buyers to launch. This approach can also make it easier for you to update your product.

For information about creating AMIs for your product, see AMI-based products (p. 61). For information about completing AWS CloudFormation templates for your product, see the section called “AMI-based Delivery Using CloudFormation” (p. 70).

When you define your serverless application, you use an AWS Serverless Application Model (AWS SAM) template that you store in the AWS Serverless Application Repository. AWS SAM is an open-source framework for building serverless applications. During deployment, AWS SAM transforms and expands the SAM syntax into AWS CloudFormation syntax. The AWS Serverless Application Repository is a managed repository for serverless applications. It makes it possible for you to store and share reusable applications so buyers can assemble and deploy serverless architectures. To create and offer this type of product, complete the following steps:

Topics
- Create a serverless application (p. 75)
- Publish your application to the repository (p. 76)
- Create the AWS CloudFormation template (p. 77)
- Submit your AWS CloudFormation templates and config files (p. 79)
- Update your AWS Serverless Application Repository application permissions (p. 79)
- Share your AMI (p. 79)
- Submit your AWS CloudFormation product with AMI and serverless application (p. 79)

AWS Marketplace reviews and validates your product before your listing is created. We will send you an email message if there are issues you must resolve before the offer is listed.

As part of fulfilling a subscription, we copy the AMIs, serverless applications, and AWS CloudFormation templates to an AWS Marketplace-owned repository in each AWS Region. When a buyer subscribes to your product, we give them access, and also notify them when you update your software.

**Create a serverless application**

Your first step is to package the AWS Lambda functions used to create your serverless application. Your application is a combination of Lambda functions, event sources, and other resources that work together to perform tasks. A serverless application can be as simple as one Lambda function or contain multiple functions with other resources, such as APIs, databases, and event source mappings.

Use the AWS SAM to define a model for your serverless application. For descriptions of property names and types, see AWS::Serverless::Application in AWSLabs on GitHub. The following is an example of an AWS SAM template with a single Lambda function and AWS Identity and Access Management (IAM) role.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: An example of SAM template with Lambda function and IAM role

Resources:
  SampleFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: 'com.sampleproject.SampleHandler::handleRequest'
      Runtime: java8
```
Publish your application to the repository

To publish an application, you first upload the application code. Store your code artifacts (for example, Lambda functions, scripts, configuration files) in an Amazon S3 bucket your account owns. When you upload your application, it is initially set to private, meaning that it is only available to the AWS account that created it. You’ll need to create an IAM policy that grants AWS Serverless Application Repository permissions to access the artifacts you uploaded.

To publish your serverless application to the serverless application repository

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose the Amazon S3 bucket you used to package your application.
3. Choose the Permissions tab.
4. Choose Bucket Policy.
5. Paste the following example policy statement. Replace `DOC-EXAMPLE-BUCKET` in the Resource property value with the bucket name for your bucket.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "serverlessrepo.amazonaws.com"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
    }
  ]
}
```
6. Choose **Save**.
8. On the **My Applications** page, choose **Publish application**.
9. Complete the required fields and any optional field, as appropriate. The required fields are:
   - Application name
   - Author
   - Description
   - Source code URL
   - SAM template
10. Choose **Publish Application**.

To publish subsequent versions of your application:
2. In the navigation pane, from **My Applications**, choose the application.
3. Choose **Publish new version**.

For more information, see **Publishing serverless Applications Using the AWS SAM CLI**.

**Create the AWS CloudFormation template**

To build your AWS CloudFormation templates, you must meet the template prerequisites and provide the required input and security parameters. For more information, see **Template Anatomy** in the **AWS CloudFormation User Guide**.

In your AWS CloudFormation template, you can reference your serverless application and your AMI. You can also use nested AWS CloudFormation templates and reference serverless applications both in the master template and the nested templates. To reference the serverless application, you use the AWS Serverless Application Model (AWS SAM) template. You can automatically generate the AWS SAM template for your application from the AWS Serverless Application Repository. The following is an example template.

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: An example master template for a SAR application
Resources:
  SampleSARApplication:
    Type: AWS::Serverless::Application
    Properties:
      Location:
        ApplicationId: arn:aws:serverlessrepo:us-east-1:1234567890:applications/TestApplication
  SampleEC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: "ami-79fd7eee"
      KeyName: "testkey"
      BlockDeviceMappings:
        - DeviceName: "/dev/sdm"
          Ebs:
```


The AWS SAM template contains the following elements.

- **ApplicationID**: Your application's ARN. This information is located in the **My Applications** section of the AWS Serverless Application Repository.

- **SemanticVersion**: The version of your serverless application. You can find this from the **My Applications** section of the AWS Serverless Application Repository.

- **Parameter (optional)**: Application parameters.

If you are planning to reference config/script files in your AWS CloudFormation template, follow the format shown below. For nested templates (AWS::Cloudformation::Stack), only TemplateURLs without intrinsic functions are supported. Note the **Parameters** content in the template.

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Metadata:
  Name: Seller test product
Parameters:
  CFTRefFilesBucket:
    Type: String
    Default: "seller-bucket"
  CFTRefFilesBucketKeyPrefix:
    Type: String
    Default: "cftsolutionFolder/additionCFfiles"
Resources:
  TestEc2:
    Type: AWS::EC2::Instance
    Metadata:
      AWS::CloudFormation::Init:
        addCloudAccount:
          files:
            /etc/cfn/set-aia-settings.sh:
              source:
                Fn::Sub:
                - https://${CFTRefFilesBucket}.${S3Region}amazonaws.com/
                ${CFTRefFilesBucketKeyPrefix}/sampleScript.sh
          S3Region:
            !If
            - GovCloudCondition
              - s3-us-gov-west-1
              - s3
            owner: root
            mode: '000700'
            authentication: S3AccessCreds
  SampleNestedStack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: 'https://sellerbucket.s3.amazonaws.com/sellerproductfolder/nestedCft.template'
      Parameters:
        SampleParameter: 'test'
    Transform: AWS::Serverless-2016-10-31
```
Submit your AWS CloudFormation templates and config files

To submit your AWS CloudFormation template and config/scripts files, grant AWS Marketplace permissions to read the Amazon S3 bucket where these files are stored. To do so, update your bucket policy to include the following permissions.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "assets.marketplace.amazonaws.com"
      },
      "Action": ["s3:GetObject", "s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                   "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"]
    }
  ]
}
```

Update your AWS Serverless Application Repository application permissions

To submit your AWS Serverless Application Repository application to AWS Marketplace, you must grant AWS Marketplace permissions to read your application. To do that, add permissions to a policy associated with your serverless application. There are two ways to update your application policy:

- Go to the AWS Serverless Application Repository. Choose your serverless application from the list. Select the Sharing tab, and choose Create Statement. On the Statement configuration page, enter the following service principal, `assets.marketplace.amazonaws.com`, in the Account Ids field. Finally choose Save.
- Use the following AWS CLI command to update your application policy.

```
aws serverlessrepo put-application-policy \
  --region region \n  --application-id application-arn \n  --statements Principals=assets.marketplace.amazonaws.com,Actions=Deploy
```

Share your AMI

All AMIs built and submitted to AWS Marketplace must adhere to all product policies. Self-service AMI scanning is available in the AWS Marketplace Management Portal. With this feature, you can initiate scans of your AMIs and receive scanning results quickly (typically, in less than an hour) with clear feedback in a single location. After your AMI has been successfully scanned, submit the AMI for processing by the AWS Marketplace Seller and Catalog Operations team by uploading your product load form.

Submit your AWS CloudFormation product with AMI and serverless application

Keep the following in mind before you submit your product:
Private images

You can use private image builds to let buyers purchase your installable software products through AWS Marketplace, and then install those products on a gold image or Amazon Machine Image (AMI) they choose from the images available to their AWS account. A gold image is a buyer-provided server image that includes a base operating system with modifications applied to help ensure the software adheres to the buyer’s IT standards. Gold images allow buyers to better meet their internal security, compliance, and management requirements.

This topic describes how to use the AWS Marketplace Management Portal (AMMP) to upload your software binaries and/or scripts and create an installable package group for each operating system (OS) your software will run on. AWS Marketplace does a test build by installing the package group on a base OS you specify and scans the resulting image for certain known vulnerabilities. After the image build and scan completes, you can use the AMMP to submit your product.

The following diagram shows the private image build flow.

1. You upload an installable software package to AWS Marketplace.
2. A buyer selects your product from AWS Marketplace, but wants to use their own gold image for the OS on the AMI.
3. The buyer requests a new AMI from the AWS Marketplace Private Image Build Service, specifying their gold image and your installable software package.

4. The AWS Marketplace Private Image Build Service creates a new private image that the buyer can use in their AWS account only. They can then launch the AMI from the private image configuration panes or within Amazon Elastic Compute Cloud (Amazon EC2).

Package group requirements

You can submit your package group for use on either AWS Marketplace base Linux AMIs or AWS Marketplace base Windows Server AMIs.

When you select the OS platform for your product, you will have the option to select multiple OSes and OS versions on which your package group will run. Windows Server packages will not run on Linux OSes and vice versa, so if you want your product to support private images for both OS platforms, you will need to define at least two package groups. When you define your package group, you upload the installation packages or scripts and AMMP will build and scan a test image for each OS you choose.

For your package group to successfully complete the build and scan process, you must adhere to these guidelines:

- The package group must have one of the packages or scripts marked as the installer. For example, the installer may be a batch file or script that orchestrates the installation of the other packages and provides the required parameters for an unattended installation.
- For Windows Server-based packages, the supported installer types have .msi, .ps1, .bat, and .exe extensions.
- For Linux/UNIX (or any POSIX-compliant) systems, the supported installer types have .exe, .rpm, .deb, .sh, and .run extensions.
- The entire installation process must be unattended. It cannot require any interactive input, and all parameters or switches must be included in the installer.
- The packages must install without downloading patches or configuration files (be complete) from another website.
- The installer/installation script must be synchronous. For example, the script must not exit until the packages are completely installed.
- The installer must exit with exit status 0 when the installation is successful. Any value other than 0 is used for unsuccessful installations.
- The installer cannot require a reboot during the installation. A reboot would stop the agent that tracks the test and scan process for packages. If your installer reboots, the agent is stopped and the test and scan will fail.
- The installer must not affect the network routing on the instance in such a way that the host becomes unreachable.

Submitting your package group

To submit a package group to AWS Marketplace for use with Private Image Build

1. From the AMMP, choose Assets, Private image build.
2. Under Manage packages, choose Start package.
3. In Enter a unique name for your package group, type the name of your product. The name must be less than 100 characters and can only contain alphanumeric characters, underscores, and dashes. Each product name associated with the AWS account used to create and publish package groups must be unique. After you've used a name (even if the build is unsuccessful), you can't use the name again. We recommend using a
naming convention with a revision number included in the file name. For example: 
[product_group_name]<product_name><version><platform><revision_number>

4. In **Select one or more packages**, select a package from the dropdown list or choose **Browse** to locate and select the package group you want to upload.

5. Under **Select supported operating system platform**, choose either **AWS Marketplace base Linux AMIs** or **AWS Marketplace base Windows AMIs**.

6. Under **Select supported operating systems**, choose all the OSes that your package group will support, and then choose **Submit**.

For each package group you submit, a build process is completed for each OS version you chose. After you submit your package group, you are redirected to the **Scan status** page, where you can check progress of the image building and scanning process for each package group.

### Scan status

After you submit your package group, you can check the current status on the **Scan status** tab. Each package group you've submitted is listed. Choose the arrow next to the package group to expand the list and show the build and scan status for each package group you selected.

Each entry will show the AMI ID, date you submitted the package group, and the status of the package group (or build). During the process, you can track the state of package groups and individual builds you have submitted. There are four states your package group submission can be in, and five states individual builds can be in.

#### Package group state

The package group state updates as automated steps complete. You can return to the **Scan status** page to check on progress, or if the page is open you can choose **Refresh status** to update the information on the page. The package group states are:

- **Building** – You have submitted your package group and the corresponding image(s) are being built.
- **Scanning** – You have submitted your package group and the corresponding images(s) are being scanned.
- **Successful** – All builds associated with your package group were successfully scanned. Submit your product load form.
- **Issues Found** – One or more builds for your submission failed that require your attention. Choose the **information** icon next to the status for additional troubleshooting information.
- **Investigating** – There was a problem uncovered during the build and scan process. AWS Marketplace is investigating.

**Note**

If your status remains in the **Investigating** state for four or more business days, contact the AWS Marketplace Seller Operations team.

#### OS build state

On the **Scan status** page you can choose the arrow next to the package group name to expand the entry to show each OS build that is part of the package group. The OS build states are:

- **Building** – The build of your software on the OS is in progress. This might take up to an hour to complete for each build.
- **Scanning** – The build process completed successfully and the scan is in progress. This might take several hours to complete.
• **Successful** – The build and scan process completed successfully. No further action on your part.
• **Issues found** – There was a problem with the build or the scan process that require your attention. Choose the **Information** icon next to the status for additional troubleshooting information.
• **Investigating** – The build or scan process failed. AWS Marketplace is investigating.

**Note**
If your status remains in the **Investigating** state for four or more business days, contact the AWS Marketplace Seller Operations team.

When your package group shows a status of **Successful**, this phase is complete. Next, you can publish your package group as a new fulfillment option for your product on AWS Marketplace.

### Submitting your product to AWS Marketplace

After you upload a package group to AWS Marketplace you can submit a product load form to publish it as a new fulfillment option for your product, or as a new product if it does not already exist. The load form is an Excel spreadsheet. The first tab of the spreadsheet provides instructions for providing the metadata needed to publish your product on AWS Marketplace.

**To download and complete the load form**

1. From the AMMP, under the **Assets** tab, choose **File upload**.
2. On the **File Uploads** page, under **Product load forms and seller guides**, choose **Private Image Form**.
3. Download the product load form.
4. Complete the form.
5. From the AMMP **Assets** tab, choose **File upload**.
6. Choose the files you want to submit and enter a brief description.

AWS Marketplace creates or update your product entry. If there are any questions on your submission, AWS Marketplace will contact you for clarification. Your product is typically added or updated within five business days.

When adding a package group as a new fulfillment option for your product, consider the following options:

• Add the package group as an additional fulfillment option to an existing software version, on an existing public product on AWS Marketplace. With this approach, the software version on the AMI and package fulfillment options must match. AWS Marketplace cannot replace an AMI on an existing software version.
• If the package group has different software than what currently exists on AWS Marketplace, you can list the package group as a new software version on an existing product. Using this approach, you must provide a successfully built and scanned AMI from the AMMP **Packages** tab. You will have the option to test package fulfillment before making the new package group public. However, the AMI will be visible to buyers right away. This is consistent with the current experience for new software versions.

### AMI product checklist

Before submitting your AMI product request to AWS Marketplace, review this checklist. Validating this information will help to make sure your submission goes through the publication process smoothly.

**Product usage:**
• Your AMI must be production-ready.
• Your AMI can't restrict product usage by time or any other measurements.
• Your AMI must be compatible with the 1-Click fulfillment experience.
• Everything required to use the product is in the software, including client applications.
• The default user uses a randomized password, or creating the initial user requires verification that the buyer is authorized to use the instance using a value unique to the instance such as instance ID.

For free or paid products:
• No additional license is required to use the product.
• The buyer doesn't have to provide personally identifiable information (for example, their email address) to use the product.

AMI preparation:
• Uses HVM virtualization and 64-bit architecture
• Doesn't contain any known vulnerabilities, malware or viruses
• Buyers have OS-level administration access to the AMI
• Run your AMI through AMI Self-Service Scanning

For Windows AMIs:
• Uses the most recent version of EC2ConfigService
• Ec2SetPassword, Ec2WindowsActivate, and Ec2HandleUserData are enabled in your AMI
• No guest accounts or remote desktop users are present

For Linux AMIs:
• Root login is locked or disabled
• No authorized keys, default passwords, or other credentials are included

Product Load Form or Product tab
• All required fields are completed
• All values are within specified character limits
• All URLs load without error
• The product image is at least 110 pixels wide and between a 1:1 and 2:1 ratio
• Pricing is specified for all enabled instance types (for hourly, hourly-based monthly pricing, and hourly-based annual pricing models)
• Monthly pricing is specified (for hourly-based monthly and monthly pricing models)
Container-based products

AWS Marketplace supports software products that use Docker containers. Container products consist of fulfillment options which are a set of container images and deployment templates that go together. You submit at least one fulfillment option for your product, with up to a maximum of four. For each fulfillment option, you provide a set of container images, usage instructions, and links to deployment templates for customers to launch that fulfillment option.

AWS Marketplace buyers see the available fulfillment options on the published product detail pages they have visibility to. Once they subscribe to the product and choose their preferred fulfillment option on the Configure your product page, they see links to the available deployment templates, container image URLs, as well as instructions for how to pull the individual container images.

You can run paid container products on the orchestration services Amazon ECS, Amazon EKS, and Fargate. You can run Free and BYOL container products on any Docker-compatible runtime.

Getting help

For assistance with your container products, contact your business development partner for AWS Marketplace or contact us.

Getting started with container products

This topic outlines how to get started with a container product, and goes through all the steps related to creating, testing, and publishing your first container product. For this exercise, we assume that you already have at least one container created in Amazon ECS, Amazon EKS, or Fargate, and that you have links for the associated images. We recommend that you plan your pricing, entitlement, and metering strategy well in advance of publicly publishing your product.

Topics

- Creating a container product (p. 85)
- Downloading and filling out the product load form for container products (p. 86)
- Integrating metering for your container product (p. 87)
- Publishing container products (p. 87)
- Container product scans (p. 88)

Creating a container product

The following procedure outlines how to create a new container product in the AWS Marketplace Management Portal.

To create a container product

1. Open a web browser and sign into the AWS Marketplace Management Portal.
2. From the menu bar, expand Assets, and choose Container.
3. Provide a customer-facing name for your product, and choose **Create**.

4. Make a note of the **Product ID** and **Product code**, as you'll need them when you fill out the product load form.

5. Choose **Create new group** to create a container group. Each container group represents a fulfillment option for your product.

6. Provide a customer-facing name for your container group. We strongly recommend that each container group within a product have a unique name.

7. In **Image location**, provide the URL for one image in this container group, and choose **Add and scan**. When you submit a container image URL, we scan it and check for security vulnerabilities. Typically, this scan takes 20 to 25 minutes. For more information, see **Container product scans** (p. 88).

   These URL links can be the name of a public external repository or a URL to a private repository such as Amazon Elastic Container Registry (Amazon ECR). For images in Amazon ECR, you must specify the image tag. You can have up to four links to deployment templates for each fulfillment option.

   For example, you could use either of the following formats as pointers to your images:

   - nginx:*mytag*
   - 123456789012.dkr.ecr.us-west-2.amazonaws.com/nginx:*mytag*

8. Repeat the previous step for each of your images in this container group. When you have finished adding images to your container group, choose **Submit container group**.

9. A dialog box will open, prompting you to finalize your container group. Once finalized, you won't be able to edit the container group again. To finalize your container group, its name, and the included images, choose **Get ID**.

10. Make a note of the container group ID, as you'll need it when you fill out the product load form.

11. (Optional) Create additional container groups. Each container product can have up to 4 container groups, one for each fulfillment option.

You've now created your container product. When you first create a container product it is not automatically published. AWS Marketplace publishes your product once you have submitted the completed product load form with your product metadata and your container images have been successfully scanned. For more information, see **Publishing container products** (p. 87).

Next, you'll need to perform two different processes, downloading and filling out the product load form, and integrate and test metering into your software. These two steps can be done in any order, or in parallel.

**Downloading and filling out the product load form for container products**

The following procedure outlines what to do with the product load form.

**To download and fill out the product load form**

1. Open a web browser and sign into the **AWS Marketplace Management Portal**.

2. From the menu bar, expand **Assets**, and choose **File Upload**.

3. From **Product load forms for download** on the right side, choose **Containers Product Load Form**.

4. Open the spreadsheet on your computer and fill out the fields to define your product. This will include your product ID and container group ID, which you made notes of when you created your container product.

   - **Columns** **Product ID** through **End User License Agreement URL** are for standard product information such as title, description, product highlights, free trials, product categories, logo image
URL, and EULA. Columns Container Group 1: Group ID through Container Group 1: Deployment Template URL 4 are for metadata specific to your first container fulfillment option. Additional fulfillment options metadata can be provided from columns Container Group 2: Group ID through Container Group 4: Deployment Template URL 4.

Tip
When viewing the product load form in Microsoft Excel, hover over each of the fields to show comments that provide guidance on how to fill in each field.

You must provide pricing and metering dimensions, based on your pricing model for your product. For more information, see the following:

- Product load form for custom metering (p. 94)
- Product load form for hourly metering (p. 99)

5. Save your product load form.

6. Open a web browser and sign into the AWS Marketplace Management Portal.

7. From the menu bar, expand Assets, and choose File Upload.

8. In Upload File, browse your computer and choose the product load form file you saved for this container product.

9. Provide a brief description for your form to help you identify it among the other product load forms you upload.


11. Your uploaded product load form will appear in a table at the bottom of the page.

Integrating metering for your container product

You use the AWS Marketplace Metering Service for both checking entitlement to use your product and metering usage for billing. For more information, see AWS Marketplace Metering Service integration (p. 91).

Publishing container products

When you publicly publish a container product, you make it visible to all AWS customers who can then subscribe and launch your product. The pricing model can't be changed for publicly published products. Before you can publish a product, you'll need to have completed the following previous steps:

- Creating a container product (p. 85)
- Downloading and filling out the product load form for container products (p. 86)
- Integrating metering for your container product (p. 87)

We will review the information in your product load form, as well as your test calls to the AWS Marketplace Metering Service. After that, we publish your product in a limited visibility state for your review and approval. As a parallel effort, you would also complete the necessary engineering integration for your product and related offers. During this time you should review your product including image links, deployment templates, descriptions, and prices to ensure accuracy.

Once all testing is complete, and you've approved the product, we publish the container product publicly. As part of the publishing process, your container images are copied to an AWS Marketplace repository on Amazon ECR. You must update the references in your deployment templates to point to the new image URLs or your product may not work as intended.

The final URL for each image will be of the following format:
Note
The account that you use is the AWS Marketplace account that the secure Amazon ECR repository is created with. That account ID can change across images and product versions.

Container product scans

When you submit a container image URL, we scan it and check for security vulnerabilities. We examine the images you provide for known security vulnerabilities. To do this, we perform a layer-by-layer static scan on the image. If we find critical vulnerabilities with remotely exploitable risk vectors, we present the list of found issues. We strongly recommend that you perform your own security analysis using a container image scanner such as Clair, Twistlock, Aqua Security, or Trend Micro to avoid delays in the ingestion and publishing process.

Your choice of base image for building your container images can have a significant influence on the security profile of the final image. If you pick a base image that already has known critical vulnerabilities, they will get flagged because of the base layer, even if your application software layers are clean. We recommend that you verify that you are starting with a vulnerability-free base container before you build your images and submit them to AWS Marketplace.

After the scan is complete, we will provide the container group ID that you need to use in the product load form to identify the set of images associated with the fulfillment option you are creating. You can define up to four fulfillment options for each container product you submit, with up to 50 container images in each set.

Container-based product requirements

AWS Marketplace maintains the following requirements for all container-based products and offerings on AWS Marketplace. These requirements help to promote a safe, secure, and trustworthy catalog for our customers. We also encourage sellers to review implementation of additional controls and protocols as applicable to meet the needs of their specific products.

All products and their related metadata are reviewed when submitted to ensure that they meet or exceed current AWS Marketplace requirements. We review and adjust these policies to meet our evolving security and other usage requirements. AWS Marketplace continuously verifies that existing products continue to meet any changes to these requirements. If products fall out of compliance, AWS Marketplace will contact you to update your product. In some cases, your product might temporarily be unavailable to new subscribers until issues are resolved.

Security requirements

All container-based products must adhere to the following security requirements:

- Docker container images must be free from any known malware, viruses, or vulnerabilities. The self-service container images ingestion tool can detect vulnerabilities. To use the container scanning tool, sign into the AWS Marketplace Management Portal, select Container from the Assets menu, and submit an image for your product.
- If your container-based products requires access to manage AWS resources, it must be achieved through IAM roles for service accounts (if run through Amazon Elastic Kubernetes Service (Amazon EKS)) or IAM roles for tasks (if run through Amazon Elastic Container Service (Amazon ECS)) instead of requesting an access key from users.
• Container-based products must only require least privileges to run. For more information, see ECS security and EKS security.
• Container images should be configured to run with non-root privileges by default.

**Access requirements**

All container-based products must adhere to the following access requirements:

• Container-based products must use an initial randomized password. Container-based products must not use initial fixed or blank passwords for external administrative access (for example, to log in to the application via a web interface). The buyer must be prompted for this randomized password before being permitted to set or change their own credentials.
• Any outside access to the application must be explicitly agreed to and enabled by customers.

**Customer information requirements**

All container-based products must adhere to the following customer information requirements:

• Container-based products must be able to be launched without requiring the buyer to provide any identifying information to activate the product. Bring Your Own License (BYOL) products are an exception to this requirement.
• Software must not require, collect, or export any customer data without the customer's knowledge and express consent.

**Product usage requirements**

All container-based products must adhere to the following product usage requirements:

• Sellers can only list fully functioning products. Beta or prerelease products for trial or evaluation purposes are not allowed. Developer, community, and BYOL editions of commercial software are supported if the seller provides an equivalent paid version on AWS Marketplace within 90 days of providing the free edition.
• All of a container-based product's usage instructions must include all steps to deploy container-based products. Usage instructions must provide commands and deployment resources pointing to the corresponding container images on AWS Marketplace.
• Container-based products must include all container images that a subscriber needs to use the software. In addition, container-based products must not require a user to launch the product using any resources from outside AWS Marketplace (for example, container images from third-party repositories).
• Product software and metadata must not contain language that redirects users to other cloud platforms, additional products, or upsell services that aren't available on AWS Marketplace.

**Architecture requirements**

All container-based products must adhere to the following architecture requirements:

• Source container images for AWS Marketplace must be provided from either a public container registry or Amazon Elastic Container Registry (Amazon ECR).
• Container images must be based on Linux.
• Paid container-based products must be able to be deployed on Amazon ECS, Amazon EKS, or AWS Fargate.

Pricing container products

This section outlines the available pricing models for container products. You can list free products, bring-your-own-license (BYOL) products, and paid products for Amazon ECS, Amazon EKS, and Fargate. You can set only one price per product.

Note
You use the AWS Marketplace Metering Service to enforce entitlement and meter usage for your paid products. For per task or per pod pricing, usage is metered automatically by AWS.

The price you set for a container product applies to all AWS Regions. Whenever you lower the price for a container product, the new price is implemented for your buyers immediately. For price increases, existing buyers are notified about the change 90 days before it impacts their billing. New buyers are billed the new amount.

Pricing models for container products

Paid container products support the following pricing models:

• Custom metered prices based off of dimensions you define (for example users, nodes, repositories, GB, etc.), up to 24 dimensions per product.
• A long term contract, at a reduced price, paid up front or in regular installments. A long term contract can be added to an existing product that has custom metered pricing, or per task / per pod pricing. Buyers pay the metered prices when they consume more than what they purchased in the long term contract.
• A fixed monthly price that provides users with unlimited product usage during the following month.
• Per Amazon ECS task or Amazon EKS pod pricing that we measure to the second with the price set per hour.
• BYOL pricing, which is managed outside of AWS Marketplace through an external billing relationship you maintain with the buyer.

Example Fixed monthly price

You set the price for your product at $99 per month. Your product includes three different container images that are deployed using an Amazon ECS task definition.

After a buyer subscribes to your product, they’re immediately charged $99, which repeats each month until they cancel the subscription. The buyer also gets unlimited usage of the product. The buyer also pays separately for any infrastructure that the tasks run on. While subscribed, they can access your container images. They can launch and run any number of containers from those images on Amazon ECS or Amazon EKS in any configuration.

If the buyer cancels their subscription in the middle of a month, they lose access to the Amazon ECR repository where AWS Marketplace stores the container images. The buyer might have pulled and stored the original images, but they can no longer pull new container image versions that you make available through AWS Marketplace. The buyer is refunded for the unused portion of the final month, and you’re paid based on the buyer’s usage minus the agreed-to AWS Marketplace fee.

Example Custom metric pricing dimensions

Your product charges by users. You have admin users and regular users and you define the pricing as $2 for admin users and $1 for regular users. You can set them up as separate dimensions when listing your product. You charge by users logged in per day and you meter that usage per day.
Example Per task or per pod hourly price

Your product includes three different container images: a controller node, a worker node, and an analytics node. Because your product isn't functional or useful without the controller node, you decide that is the image that you want to charge usage for. You set a price of $6 per hour.

You modify the software in the container image for the controller node to integrate with the AWS Marketplace Metering Service's `RegisterUsage` operation. This ensures that only buyers with an active subscription can launch and run that container image and that its usage is metered based on how long it runs.

The buyer is charged $6 per hour of usage for each Amazon EKS controller pod running. If the buyer launches five Amazon EKS controller pods that include the controller node container, they're charged $30 per hour ($6 per pod). The buyer also pays separately for any infrastructure that the pods run on.

For hourly pricing, billing is per-second with a 1-minute minimum. If the customer runs this controller container for 20 minutes and 30 seconds, they're charged $20 \times \left( \frac{$6}{60} \right) + 30 \times \left( \frac{$6}{60/60} \right) = \$2 + \$0.05 = \$2.05$. You're paid based on the buyer's usage minus the agreed-to AWS Marketplace fee.

Example Long term contracts

For metered pricing models, you can add a long term contract price for buyers to get a discount for committing upfront. Say that the buyer commits to pay for regular users upfront for a one year contract, and the price drops from $1 per user to $0.5 per user.

For the per task / per pod example, you can drop the price from $6 per pod to $3 per pod if they commit upfront to running those pods for a year.

In both cases, buyers that purchase long term contracts will be billed upfront, either as a one-time payment or regularly scheduled future payments. Buyers will also be billed for any additional usage above their contract at the metered rate.

AWS Marketplace Metering Service integration

You use the **AWS Marketplace Metering Service** for both checking entitlement to use your product and metering usage for billing. If you want to define your own pricing units and meter that usage to us for billing, integrate with **MeterUsage**. If you want to price your product based on number of tasks or pods used and have us meter that usage automatically, integrate with the **RegisterUsage** action. For both types of pricing, you can add a long term contract price without changing how you integrate with the AWS Marketplace Metering Service.

When you create a new container product in AWS Marketplace Management Portal, we provide a set of product identifiers (the product code and public key) that are used to integrate your product with the AWS Marketplace Metering service.

Entitlement

Integrating with the AWS Marketplace Metering Service allows you to verify that the customer running your paid software is subscribed to your product on AWS Marketplace, guarding you against unauthorized use at container startup. You call the **MeterUsage** or **RegisterUsage** action, depending on your pricing model, to verify entitlement. For hourly and fixed monthly pricing models, use the **RegisterUsage** action. For custom metering pricing models, use the **MeterUsage** action.

If a buyer is not entitled to your product, either one of these API actions will return the **CustomerNotEntitledException** exception.
Note
If a buyer unsubscribes from your product while running it, they are entitled to continue running it. However, they can't launch additional containers for your product.

Integration guidelines

Keep the following guidelines in mind as you create and publish your container products and plan to use the AWS Marketplace Metering Service `MeterUsage` or `RegisterUsage` actions for entitlement and metering.

- Do not configure AWS credentials within your software or the Docker container image. AWS credentials for the buyer are automatically obtained at runtime when your container image is running within an Amazon ECS task or Amazon EKS pod.
- To call `MeterUsage` or `RegisterUsage` from Amazon EKS, you must use a supported AWS SDK.
  To test `MeterUsage` or `RegisterUsage` integration of Amazon EKS, you must run an Amazon EKS cluster running Kubernetes 1.13.x or greater. Kubernetes 1.13 is required for IAM roles for pod support, which is a dependency for the running pod to obtain the AWS credentials required to invoke these actions on Amazon EKS.
- You can do local development, but you will get a `PlatformNotSupportedException` exception. This exception won't occur when you launch the container on AWS container services (Amazon ECS, Amazon EKS, and Fargate).

Supported AWS Regions

For a list of all AWS Marketplace supported AWS Regions, see Region Table on the Global Infrastructure page.

Obtaining the AWS Region for metering

When integrating your container with either `MeterUsage` or `RegisterUsage` for metering, don't configure the AWS SDK to use a specific AWS Region. The AWS Region must be obtained dynamically at runtime. For example, if a customer launches an Amazon ECS task or Amazon EKS pod, and `RegisterUsage` is called in an AWS Region that differs from that of where the Amazon ECS task or Amazon EKS pod was launched, `RegisterUsage` will throw `InvalidRegionException`.

AWS SDK languages do not determine the `AWS_REGION` in a consistent manner. For example, the AWS SDK for Java automatically uses `Amazon EC2 instance metadata` (specifically, `ec2InstanceMetadata`) to obtain the AWS Region when environment variables or other configuration are not present. In this instance, only call `ec2InstanceMetadata` if the `AWS_REGION` environment variable isn't present.

For information on how to dynamically obtain an AWS Region at runtime, refer to the AWS SDK Developer Guide for your programming language.

Preventing metering modification

Introducing ways for buyers to modify or override calls to `RegisterUsage` or `MeterUsage` might result in undesirable billing and payment issues. We strongly recommend that you integrate the metering and entitlement logic.

Keep the following in mind when engineering your product to prevent metering modification:

- If buyers can insert new image layers that contain CMD or ENTRYPOINT instructions, directly integrate `RegisterUsage` or `MeterUsage` into the software the buyer is running through your container image. Otherwise, calls to `RegisterUsage` or `MeterUsage` executed via CMD or ENTRYPOINT from the base image will likely be overridden by the buyer.
• We recommend that you manage the AWS Marketplace product codes your software uses as input to `RegisterUsage` or `MeterUsage` in a manner buyers can't modify. However, if your product manages product codes in a manner customers can override, such as AWS CloudFormation, Helm chart, or Kubernetes manifest, you must maintain a list of trusted AWS Marketplace product codes to ensure the one your software passes as input to `RegisterUsage` or `MeterUsage` is valid.

• If any of your trusted product codes are for free products, ensure that they can't be used in place of a paid product code.

**Custom metering**

AWS Marketplace container products can have custom metering on up to 24 different pricing dimensions per product. Each dimension can have a long-term contract price associated with it. Custom metering is enabled by integrating your container product with the AWS Marketplace Metering Service. If you want to define your own pricing units and custom metering for that usage to us for billing, integrate with `meterUsage`.

Price dimensions are defined in two locations, once in the product load form and once through the `MeterUsage` operation. This two-factor method ensures that the subsequent offers are working as intended before they're made available to the public.

To set up custom metering, you'll need to choose the usage category, the unit type, and pricing dimensions:

• **Usage category** – The usage category helps buyers understand what your product is and how to use it.

• **Unit type** – The unit type defines the unit of measure for billing. For example, bandwidth measured in GBps or MBps, the number of hosts, or data measured in MB, GB, or TB.

• **Pricing dimensions** – The pricing dimensions represents a feature or service that you've set a per-unit price for (for example, users, scans, vCPUs, or deployed agents). Pricing dimensions are public; however, you can still define private and Bring Your Own License (BYOL) offers for public products. Don't send pricing in the metering records. You meter the quantity of units, and we use that along with the prices you defined when creating your product to compute the buyer's bill.

If your product pricing doesn't fit with any of the predefined categories or unit types, you can choose the generic units category and use the dimension description to describe what the unit is.

Optionally, you may distribute the usage into allocations by properties that you track. The allocations are represented as tags to the buyer. These tags allow the buyer to view their costs split into usage by tag values. For example, if you charge by the user, and users have a "Department" property, you could create usage allocations with tags that have a key of "Department", and one allocation per value. This does not change the price, dimensions, or the total usage that you report, but allows your customer to view their costs by categories appropriate to your product.

We recommend that you send a metering record every hour; however, you can aggregate usage over daily or monthly periods as well. If you experience an outage you can aggregate buyer software use and send it in the following hours metering. You can't send more than one record per hour.

**Important**

Free trial and prepaid entitlement are tracked on an hourly level. As a result, sending these records in separately might lead to the buyer being overcharged.

**Custom metering prerequisites**

Before publishing the product, you must do the following:

2. Fill out the product load form with the necessary dimension information, and return it to us for processing.

3. Use an AWS Identity and Access Management (IAM) role for the task or pod running your application with the IAM permissions necessary to call `MeterUsage`. The IAM managed policy `AWSMarketplaceMeteringRegisterUsage` has these permissions.

4. (Optional) We recommend that you enable CloudTrail logging in the task or pod definition if you want to see logging.

5. Make a test call to the `MeterUsage` operation with a record for all of the pricing dimensions you define.

### Product load form for custom metering

When filling out the product load form for custom metering, each product can have up to 24 dimensions. The dimensions are defined in the following fields:

- **Dimension Name** – The name used when your container application is sending metering records to the AWS Marketplace Metering Service. This name indicates which dimension your buyer will use. This name is visible in billing reports. After you set the name, you can't change it.

- **Dimension Description** – The buyer-facing description for the dimension. The description can't exceed 70 characters. After the product is published publicly to buyers, this field can't be changed.

- **Dimension Rate** – The software price per unit for this product when buyers pay as they go. This field supports three decimal places.

- **Dimension Long Term Rate** – The total software price over a long-term contract when buyers pay upfront.

- **Long Term Duration (Days)** – The duration, in days, for the long-term contract.

### Testing MeterUsage integration and preview mode

Use the `MeterUsage` operation to test your integration before submitting your image to AWS Marketplace for publishing.

Preview mode operates identically to production mode, except preview mode does not verify entitlement to use your product. To call `MeterUsage` in preview mode, call `MeterUsage` from the container image(s) by running your product on Amazon Elastic Container Service (Amazon ECS) or Amazon Elastic Kubernetes Service (Amazon EKS) with the AWS account you are using to list the product on AWS Marketplace. When testing, launch at least one Amazon ECS task or Amazon EKS pod containing your paid container in the US East (N. Virginia) AWS Region.

**Note**

If your product supports both Amazon ECS and Amazon EKS, you only need to launch in Amazon EKS for us to validate your integration.

You can't fully test the integration until your product is published with all the required metadata and pricing information. If requested, the AWS Marketplace catalog operations team can verify receipt of your metering records in preview mode.

### Error handling for MeterUsage

If your container image integrates with the `MeterUsage` operation and receives an exception other than `ThrottlingException` at container startup, you should terminate the container to prevent unauthorized use.

Exceptions other than `ThrottlingException` are thrown only on the initial call to `MeterUsage`. Subsequent calls from the same Amazon ECS task or Amazon EKS pod do not throw...
CustomerNotSubscribedException even if the customer unsubscribes while the task or pod is still running. These customers are still charged for running containers after they unsubscribe and their usage is tracked.

The following table describes the errors that MeterUsage might throw. Each AWS SDK programming language has a set of error handling guidelines that you can refer to for additional information.

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DuplicateRequestException</td>
<td>A metering record has already been emitted for the given {usageDimension, timestamp} with a different usageQuantity.</td>
</tr>
<tr>
<td>InvalidUsageDimensionException</td>
<td>The usage dimension does not match one of the UsageDimensions associated with the product.</td>
</tr>
<tr>
<td>TimestampOutOfBoundsException</td>
<td>The timestamp value passed in the MeterUsage is out of allowed range.</td>
</tr>
<tr>
<td>InternalServiceErrorException</td>
<td>MeterUsage isn't available.</td>
</tr>
<tr>
<td>CustomerNotEntitledException</td>
<td>The customer doesn't have a valid subscription for the product.</td>
</tr>
<tr>
<td>InvalidProductCodeException</td>
<td>The ProductCode value passed in as part of the request doesn't exist.</td>
</tr>
<tr>
<td>ThrottlingException</td>
<td>The calls to MeterUsage are throttled.</td>
</tr>
<tr>
<td>InvalidEndpointRegionException</td>
<td>MeterUsage must be called in the same AWS Region that the Amazon ECS task or Amazon EKS pod was launched in. This prevents a container from choosing a Region (for example, withRegion(&quot;us-east-1&quot;)) when calling MeterUsage.</td>
</tr>
<tr>
<td>InvalidTagException</td>
<td>The tags in the UsageAllocations object must have unique keys within each UsageAllocation, have no key-value pairs the same across UsageAllocation objects, and the number of tags must be 5 or less per UsageAllocation.</td>
</tr>
<tr>
<td>InvalidUsageAllocationsException</td>
<td>The UsageAllocation object allocations do not add up to the UsageQuantity value.</td>
</tr>
</tbody>
</table>

Integrating your container product with the AWS Marketplace Metering Service using the AWS SDK for Java

The following example outlines an implementation that uses the AWS SDK for Java to integrate with the AWS Marketplace Metering Service MeterUsage operation. For complete details, see MeterUsage Java examples (p. 97). Many of the following steps apply regardless of the language.

**Example: AWS Marketplace Metering Service integration**

2. From **Assets**, choose **Containers** to start creating a new container product. Creating the product generates the product code for the product to integrate with your container image. For more
information about publishing, see Publishing container products (p. 87). For information about setting AWS Identity and Access Management (IAM) permissions, see the section called “AWS Marketplace metering and entitlement API permissions” (p. 211).

3. Download the public AWS Java SDK.

   **Important**
   To call the metering APIs from Amazon Elastic Kubernetes Service (Amazon EKS), you must use a supported AWS SDK and run on an Amazon EKS cluster running Kubernetes 1.13 or later.

4. Call the `MeterUsage` operation from the task or pod once every hour for each dimension usage. The API operation accepts one metering record for a unique combination of `Dimension`, `Resource`, and `Hour`. The resource is either an Amazon Elastic Container Service (Amazon ECS) task or an Amazon EKS pod.

   ```json
   
   {  
   "ProductCode": "string", // (required)  
   "UsageDimension": "string", // (required)  
   "UsageQuantity": int, // (optional) Default is 0. Acceptable value from [0,2147483647 (INT_MAX)]  
   "Timestamp": Date, // (required) Timestamp in UTC. Value can be one hour in the past.  
   "UsageAllocations": List<UsageAllocation> // (optional) UsageAllocations across 1 or more tags.  
   }
   
   ```

5. Rebuild a new version of your Docker container image that includes the `MeterUsage` call, tag the container, and push it to any Docker registry that is compatible with Amazon ECS or Amazon EKS, such as Amazon Elastic Container Registry (Amazon ECR) or Docker Hub. If you are using Amazon ECR, ensure that the account launching the Amazon ECS task or Amazon EKS pod has permissions on the Amazon ECR repository. Otherwise, the operation fails.

   **Note**
   If you use a private Docker Hub repository, follow the steps in Private Registry Authentication for Tasks in the Amazon Elastic Container Service Developer Guide.

6. Create an IAM role that grants permission for your container to call `MeterUsage`, as defined in the following code example. You must supply this IAM role in the Task Role parameter of the Amazon ECS task or Amazon EKS pod definition.

   ```json
   
   {  
   "Version": "2012-10-17",  
   "Statement": [  
   {  
   "Action": [  
   "aws-marketplace:MeterUsage"  
   ],  
   "Effect": "Allow",  
   "Resource": "*"  
   }  
   ]  
   }
   
   ```

7. Create an Amazon ECS task or Amazon EKS pod definition that references the container that has integrated with AWS Marketplace and references the IAM role that you created in step 6. If you want to see logging, enable AWS CloudTrail logging in the task definition.

8. Create an Amazon ECS or Amazon EKS cluster to run your task or pod. For more information about creating an Amazon ECS cluster, see Creating a Cluster in the Amazon Elastic Container Service Developer Guide. For more information about creating an Amazon EKS cluster (using Kubernetes version 1.1.3.x or later), see Creating an Amazon EKS Cluster.
9. Configure the Amazon ECS or Amazon EKS cluster and launch the Amazon ECS task definition or Amazon EKS pod that you created in step 8, in the us-east-1 AWS Region. It's only during this testing process, before the product is live, that you have to use this Region.

10. When you get a valid response from MeterUsage for each of the dimensions being published for the product, you can begin creating your container product. For questions, contact the AWS Marketplace Seller Operations team.

MeterUsage Java examples

The following code examples use the AWS SDK for Java and AWS Marketplace Metering Service to call the MeterUsage operation.

The following code example calls the MeterUsage operation without any UsageAllocations.

```java
import com.amazonaws.services.marketplacemetering.AWSMarketplaceMetering;
import com.amazonaws.services.marketplacemetering.AWSMarketplaceMeteringClientBuilder;
import com.amazonaws.services.marketplacemetering.model.MeterUsageRequest;
import com.amazonaws.services.marketplacemetering.model.MeterUsageResult;
import java.util.Date;

public class MeterUsage {
    private static final String PRODUCT_CODE = ".......";
    private final AWSMarketplaceMetering awsMarketplaceMetering;

    public MeterUsage() {
        awsMarketplaceMetering = AWSMarketplaceMeteringClientBuilder.standard().build();
    }

    /**
     * Submits metering record for a FCP Dimension. The API accepts 1 metering record per dimension
     * for a given buyer's resource for a given timestamp hour. Ex. If a buyer is running 10 tasks,
     * the API will accepts 1 call to MeterUsage in an hour for a given dimension for each running task.
     *
     * @param dimension - FCP dimension name provided during the publishing of the product.
     * @param quantity - FCP dimension consumption value for the hour.
     * @param timestamp - Timestamp, in UTC, for which the usage is being reported.
     *                  Timestamp cant be more than 1 hour in the past.
     *                  Make sure the timestamp value is not before the start of the software usage.
     * @param productCode - Product code.
     *
     * @return MeterUsage result
     */
    public void callMeterUsage(String dimension, int quantity, Date timestamp) {
        MeterUsageRequest meterUsageRequest = new MeterUsageRequest()
            .withProductCode(PRODUCT_CODE)
            .withUsageDimension(dimension)
            .withUsageQuantity(quantity)
            .withTimestamp(timestamp);
        MeterUsageResult meterUsageResult =
            awsMarketplaceMetering.meterUsage(meterUsageRequest);
    }
}
```

The following code example calls the MeterUsage operation with UsageAllocations.

```java
private static String callMeterUsageWithAllocationsByTag(AWSMarketplaceMetering marketplaceMetering) {
    // Tag Keys for the product
    String tagKey1 = "Key1";
```
String tagKey2 = "Key2";
String tagKey3 = "Key3";

// 1st Usage Allocation bucket which has two Tags [{Key1, Key1Value1},{Key2, Key2Value1}]
List<Tag> tagsForUsageAllocation1 = Arrays.asList(new Tag().withKey(tagKey1).withValue("Key1Value1"),
new Tag().withKey(tagKey2).withValue("Key2Value1"));
UsageAllocation usageAllocation1 = new UsageAllocation()
  .withTags(tagsForUsageAllocation1)
  .withAllocatedUsageQuantity(20);

// 2nd Usage Allocation bucket which has two Tags [{Key1, Key1Value2},{Key2, Key2Value1}]
List<Tag> tagsForUsageAllocation2 = Arrays.asList(new Tag().withKey(tagKey1).withValue("Key1Value2"),
new Tag().withKey(tagKey2).withValue("Key2Value1"));
UsageAllocation usageAllocation2 = new UsageAllocation()
  .withTags(tagsForUsageAllocation2)
  .withAllocatedUsageQuantity(20);

// 3rd Usage Allocation bucket which has two Tags [{Key1, Key1Value2},{Key2, Key2Value2},{Key3, Key3Value1}]
List<Tag> tagsForUsageAllocation3 = Arrays.asList(new Tag().withKey(tagKey1).withValue("Key1Value2"),
new Tag().withKey(tagKey2).withValue("Key2Value2"),
new Tag().withKey(tagKey3).withValue("Key3Value1"));
UsageAllocation usageAllocation3 = new UsageAllocation()
  .withTags(tagsForUsageAllocation3)
  .withAllocatedUsageQuantity(15);

// 4th Usage Allocation bucket with no tags
UsageAllocation usageAllocation4 = new UsageAllocation()
  .withAllocatedUsageQuantity(15);

List<UsageAllocation> usageAllocationList = Arrays.asList(usageAllocation1,
usageAllocation2,
usageAllocation3,
usageAllocation4);

MeterUsageRequest meterUsageRequest = new MeterUsageRequest()
  .withProductCode("TestProductCode")
  .withUsageDimension("Dimension1")
  .withTimestamp(new Date())
  .withUsageQuantity(70)
  .withUsageAllocations(usageAllocationList);

MeterUsageResult meterUsageResult;
try {
  meterUsageResult = marketplaceMetering.meterUsage(meterUsageRequest);
} catch (Exception e) {
  // Log Error
  throw e;
}
return meterUsageResult.getMeteringRecordId();

Hourly metering

If your container product uses per-hour task/pod pricing instead of custom metered pricing dimensions,
you don’t have to define custom metering dimensions.
RegisterUsage meters software use per Amazon Elastic Container Service (Amazon ECS) task or per Amazon Elastic Kubernetes Service (Amazon EKS) pod, per hour, with usage prorated to the second. A minimum of 1 minute of usage applies to tasks or pods that are short lived. Continuous metering for software use is automatically handled by the AWS Marketplace Metering Control Plane. Your software is not required to perform any metering specific actions except calling RegisterUsage once for metering of software use to commence.

The AWS Marketplace Metering Control Plane continues to bill customers for running Amazon ECS tasks and Amazon EKS pods, regardless of the customer’s subscription state, removing the need for your software to perform entitlement checks after the initial successful launch of the task or pod.

Hourly metering prerequisites

Before publishing the product, you must do the following:

2. Fill out the product load form with the necessary hourly price information and return it to us for processing.
3. Use an AWS Identity and Access Management (IAM) role for the task or pod running your application with the IAM permissions necessary to call RegisterUsage. The IAM managed policy AWSMarketplaceMeteringRegisterUsage has these permissions.
4. (Optional) We recommend that you enable AWS CloudTrail (CloudTrail) logging in the task or pod definition if you want to see logging.
5. Make a test call to the RegisterUsage action with a record for all of the pricing dimensions you define.

Product load form for hourly metering

When filling out the product load form for hourly metering, fill out the following fields for your product, in addition to the other required and optional fields that define your product:

- **Hourly Price** This is the price for your product, per hour.
- **Dimension Long Term Rate** – The total software price over a long-term contract when buyers pay upfront.
- **Long Term Duration (Days)** – The duration, in days, for the long-term contract.

Testing integration and preview mode for RegisterUsage

Use the RegisterUsage action to test your integration before submitting your image to AWS Marketplace for publishing.

Preview mode operates identically to production mode, except preview mode does not verify entitlement to use your product. To call RegisterUsage in preview mode, call RegisterUsage from the container image(s) by running your product on Amazon ECS or Amazon EKS with the AWS account you are using to list the product on AWS Marketplace. When testing, launch at least one Amazon ECS task or Amazon EKS pod containing your paid container in the US East (N. Virginia) AWS Region.

**Note**

If your product supports both Amazon ECS and Amazon EKS, you only need to launch in Amazon EKS for us to validate your integration.

You can't fully test the integration until your product is published with all the required metadata and pricing information. If requested, the AWS Marketplace catalog operations team can verify receipt of your metering records in preview mode.
Error handling for RegisterUsage

If your container image integrates with the AWS Marketplace Metering service and receives an exception other than ThrottlingException at container startup, you should terminate the container to prevent unauthorized use.

Exceptions other than ThrottlingException are thrown only on the initial call to RegisterUsage. Subsequent calls from the same Amazon ECS task or Amazon EKS pod do not throw CustomerNotSubscribedException even if the customer unsubscribes while the task or pod is still running. These customers are still charged for running containers after they unsubscribe and their usage is tracked.

The following table describes the errors that RegisterUsage might throw. Each AWS SDK programming language has a set of error handling guidelines that you can refer to for additional information.

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InternalServiceErrorException</td>
<td>RegisterUsage isn't available.</td>
</tr>
<tr>
<td>CustomerNotEntitledException</td>
<td>The customer doesn't have a valid subscription for the product.</td>
</tr>
<tr>
<td>InvalidProductCodeException</td>
<td>The ProductCode value passed in as part of the request doesn't exist.</td>
</tr>
<tr>
<td>InvalidPublicKeyException</td>
<td>The PublicKeyVersion value passed in as part of the request doesn't exist.</td>
</tr>
<tr>
<td>PlatformNotSupportedException</td>
<td>AWS Marketplace doesn't support metering usage from the underlying platform. Only Amazon ECS, Amazon EKS, and AWS Fargate are supported.</td>
</tr>
<tr>
<td>ThrottlingException</td>
<td>The calls to RegisterUsage are throttled.</td>
</tr>
<tr>
<td>InvalidRegionException</td>
<td>RegisterUsage must be called in the same AWS Region that the Amazon ECS task or Amazon EKS pod was launched in. This prevents a container from choosing a Region (for example, withRegion(&quot;us-east-1&quot;)) when calling RegisterUsage.</td>
</tr>
</tbody>
</table>

Integrating your container product with the AWS Marketplace Metering Service using the AWS SDK for Java

The following steps outline an example implementation using the AWS SDK for Java to integrate with the AWS Marketplace Metering Service's RegisterUsage action. For the full source code, see RegisterUsage Java example (p. 102). Many of these steps apply regardless of the language.

Example steps for AWS Marketplace Metering Service integration

2. From Assets choose Containers to start creating a new container product. Creating the product generates the product code for the product to integrate with your container image. For more information about publishing, see Publishing container products (p. 87). For information about setting IAM permissions, see the section called “AWS Marketplace metering and entitlement API permissions” (p. 211).
3. Download the public AWS Java SDK.

**Important**
To call the metering APIs from Amazon EKS, you must use a supported AWS SDK and run on an Amazon EKS cluster running Kubernetes 1.13 or later.

4. (Optional) If you're integrating with the `RegisterUsage` action and you want to perform digital signature verification, you need to configure the BouncyCastle signature verification library in your application classpath.

If you want to use JSON Web Token (JWT), you must also include JWT Java libraries in your application classpath. Using JWT provides a simpler approach to signature verification but is not required, and you can use standalone BouncyCastle instead. Whether you use JWT or BouncyCastle, you need to use a build system such as Maven to include transitive dependencies of BouncyCastle or JWT in your application classpath.

```
// Required for signature verification using code sample
<dependency>
  <groupId>org.bouncycastle</groupId>
  <artifactId>bcpkix-jdk15on</artifactId>
  <version>1.60</version>
</dependency>

// This one is only required for JWT
<dependency>
  <groupId>com.nimbusds</groupId>
  <artifactId>nimbus-jose-jwt</artifactId>
  <version>6.0</version>
</dependency>
```

5. Call `RegisterUsage` from each paid container image in your product offering. `ProductCode` and `PublicKeyVersion` are required parameters, and all other inputs are optional. The following is an example payload for `RegisterUsage`.

```
{
  "ProductCode": "string", // (required)
  "PublicKeyVersion": 1,    // (required)
  "Nonce": "string",       // (optional) to scope down the registration
                           // to a specific running software
                           // instance and guard against
                           // replay attacks
}
```

6. `RegisterUsage` generates an RSA-PSS digital signature using SHA-256 that you can use to verify request authenticity. The signature includes the following fields: `ProductCode`, `PublicKeyVersion`, and `Nonce`. To verify the digital signature, you must retain these fields from the request. The following code is an example response to a `RegisterUsage` call.

```
{
  "Signature": "<<JWT Token>>"
}
```

// Where the JWT Token is composed of 3 dot-separated, // base-64 URL Encoded sections. // e.g. eyJhbGcVCJ9.eyJzdWIMzkwMjJ9.rrO9Qw0SXRWe
```
```
// Section 1: Header/Algorithm
{
  "alg": "PS256",
  "typ": "JWT"
}
```
7. Rebuild a new version of your Docker container image that includes the RegisterUsage call, tag the container, and push it to any Docker registry that is compatible with Amazon ECS or Amazon EKS, such as Amazon ECR or Docker Hub. If you are using Amazon ECR, ensure that the account launching the Amazon ECS task or Amazon EKS pod has permissions on the Amazon ECR repository. Otherwise, execution fails.

**Note**

If you use a private Docker Hub repository, follow the steps in [Private Registry Authentication for Tasks](https://docs.aws.amazon.com/elasticcontainerservice/latest/developerguide/ecs-tasks-private-docker.html) in the *Amazon Elastic Container Service Developer Guide*.

8. Create an IAM role that grants permission for your container to call RegisterUsage, as defined in the following code. You must supply this IAM role in the Task Role parameter of the Amazon ECS task or Amazon EKS pod definition.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "aws-marketplace:RegisterUsage"
      ],
      "Effect": "Allow",
      "Resource": "*
    }
  ]
}
```

9. Create an Amazon ECS task or Amazon EKS pod definition that references the container that has integrated with AWS Marketplace and references the IAM role that you created in step 7. You should enable AWS CloudTrail logging in the task definition if you want to see logging.

10. Create an Amazon ECS or Amazon EKS cluster to execute your task or pod. For more information about creating an Amazon ECS cluster, see [Creating a Cluster](https://docs.aws.amazon.com/elasticcontainerservice/latest/developerguide/ecs-tasks-private-docker.html) in the *Amazon Elastic Container Service Developer Guide*. For more information about creating an Amazon EKS cluster (using Kubernetes version 1.1.3.x or later), see [Creating an Amazon EKS Cluster](https://docs.aws.amazon.com/eks/latest/userguide/cluster.creation.deployment.html).

11. Configure the Amazon ECS or Amazon EKS cluster and launch the Amazon ECS task definition or Amazon EKS pod that you created, in the us-east-1 AWS Region. It's only during this testing process, before the product is live, that you have to use this region.

12. When you get a valid response back from RegisterUsage, you can begin creating your container product. For questions, contact the AWS Marketplace Seller Operations team.

### RegisterUsage Java example

The following example uses the AWS SDK for Java and AWS Marketplace Metering Service to call the RegisterUsage operation. Signature verification is optional, but if you want to perform signature verification, you must include the required digital signature verification libraries. This example is for illustrative purposes only.

```java
import com.amazonaws.auth.PEM;
import com.amazonaws.services.marketplacemetering.AWSMarketplaceMetering;
```
import com.amazonaws.services.marketplacemetering.AWSMarketplaceMeteringClientBuilder;
import com.amazonaws.services.marketplacemetering.model.RegisterUsageRequest;
import com.amazonaws.services.marketplacemetering.model.RegisterUsageResult;
import com.amazonaws.util.json.Jackson;
import com.fasterxml.jackson.databind.JsonNode;
import com.nimbusds.jose.JWSObject;
import com.nimbusds.jose.JWSVerifier;
import com.nimbusds.jose.JWSVerifier;
import com.nimbusds.jose.crypto.RSASSAVerifier;
import java.io.ByteArrayInputStream;
import java.nio.charset.StandardCharsets;
import java.security.PublicKey;
import java.security.Security;
import java.security.Signature;
import java.security.interfaces.RSAPublicKey;
import java.util.Base64;
import java.util.Optional;
import java.util.UUID;
import org.bouncycastle.jce.provider.BouncyCastleProvider;

/**
 * Class for making calls out to AWS Marketplace Metering Service.
 */
class RegisterUsage {
    private static final String PRODUCT_CODE = ".......";

    private final AWSMarketplaceMetering registerUsageClient;
    private final SignatureVerifier signatureVerifier;
    private final int publicKeyVersion;

    public RegisterUsage(final SignatureVerifier signatureVerifier) {
        this.signatureVerifier = signatureVerifier;
        this.publicKeyVersion = PublicKeyProvider.PUBLIC_KEY_VERSION;
        this.registerUsageClient = AWSMarketplaceMeteringClientBuilder.standard().build();
    }

    /**
     * Shows how to call RegisterUsage client and verify digital signature.
     */
    public void callRegisterUsage() {
        RegisterUsageRequest request = new RegisterUsageRequest()
            .withProductCode(PRODUCT_CODE)
            .withPublicKeyVersion(publicKeyVersion)
            .withNonce(UUID.randomUUID().toString());

        // Execute call to RegisterUsage (only need to call once at container startup)
        RegisterUsageResult result = this.registerUsageClient.registerUsage(request);

        // Verify Digital Signature w/o JWT
        boolean isSignatureValid = this.signatureVerifier.verify(request, result);
        if (!isSignatureValid) {
            throw new RuntimeException("Revoke entitlement, digital signature invalid.");
        }
    }
}

/**
 * Signature verification class with both a JWT-library based verification
 * and a non-library based implementation.
 */
class SignatureVerifier {
    private static BouncyCastleProvider BC = new BouncyCastleProvider();

    private static final String SIGNATURE_ALGORITHM = "SHA256withRSA/PSS";

    private final PublicKey publicKey;
public SignatureVerifier(PublicKeyProvider publicKeyProvider) {
    this.publicKey = publicKeyProvider.getPublicKey().orElse(null);
    Security.addProvider(BC);
}

/**
 * Example signature verification using the NimbusJOSEJWT library to verify the JWT Token.
 *
 * @param request RegisterUsage Request.
 * @param result  RegisterUsage Result.
 * @return true if the token matches.
 */
public boolean verifyUsingNimbusJOSEJWT(final RegisterUsageRequest request, final RegisterUsageResult result) {
    if (!getPublicKey().isPresent()) {
        return false;
    }
    try {
        JWSVerifier verifier = new RSASSAVerifier((RSAPublicKey) getPublicKey().get());
        JWSObject jwsObject = JWSObject.parse(result.getSignature());
        return jwsObject.verify(verifier) && validatePayload(jwsObject.getPayload().toString(), request, result);
    } catch (Exception e) {
        // log error
        return false;
    }
}

/**
 * Validate each value in the returned payload matches values originally
private boolean validatePayload(final String payload, final RegisterUsageRequest request, final RegisterUsageResult result) {
    try {
        JsonNode payloadJson = Jackson.getObjectMapper().readTree(payload);
        boolean matches = payloadJson.get("productCode").asText().equals(request.getProductCode());
        matches = matches && payloadJson.get("nonce").asText().equals(request.getNonce());
        return matches && payloadJson.get("publicKeyVersion").asText().equals(String.valueOf(request.getPublicKeyVersion()));
    } catch (Exception ex) {
        // log error
        return false;
    }
}

private Optional<PublicKey> getPublicKey() {
    return Optional.ofNullable(this.publicKey);
}

/**
 * Public key provider taking advantage of the AWS PEM Utility.
 */

class PublicKeyProvider {
    // Replace with your public key. Ensure there are new-lines ("
    // string after "-----BEGIN PUBLIC KEY-----
    private static final String PUBLIC_KEY = "-----BEGIN PUBLIC KEY-----
" + "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDdlatRjRjogo3WojgGHFHYLugd
" + "UWAY9iR3fy4arWNA1Ko58kWv3jcJibXr8bvwJAUparCw1vdbH6dvZOFouO/gCFQw
" + "HUFqrSDv+MzSUMMAe8jeKB4qW+jK+QU9a03GUnKkkle+QpX/g6jXi7r1/xAK5D
" + "o2kJ+X5x9cipRgKxIDAQAB"
" + "-----END PUBLIC KEY-----";
    
    public static final int PUBLIC_KEY_VERSION = 1;

    public Optional<PublicKey> getPublicKey() {
        try {
            return Optional.of(PEM.readPublicKey(new ByteArrayInputStream(PUBLIC_KEY.getBytes(StandardCharsets.UTF_8))));
        } catch (Exception e) {
            // log error
            return Optional.empty();
        }
    }
}
Machine learning products

AWS Marketplace enables sellers to create and provide machine learning algorithms and model packages using Amazon SageMaker. Sellers package their products as Docker containers, upload them to Amazon Elastic Container Registry (Amazon ECR), create the algorithm or model packages in Amazon SageMaker, and add them as free or paid products in AWS Marketplace.

AWS customers can find these products through the Amazon SageMaker console or AWS Marketplace, and deploy them on Amazon SageMaker. They can review product descriptions, documentation, customer reviews, pricing, and support information. When the buyers subscribe to an algorithm or model package, the product is added to their list of products on the Amazon SageMaker console. They can also use the Amazon SageMaker SDK, the AWS Command Line Interface (AWS CLI), or the Amazon SageMaker console to create a fully managed inference endpoint. Buyers can access models only through the RESTful endpoints.

For support with creating machine learning products with Amazon SageMaker, contact AWS Marketplace Seller Operations.

Getting started with Amazon SageMaker

If you are new to Amazon SageMaker, the following webinars can get you started:

- Introducing Amazon SageMaker (Level 200)
- Introducing Amazon SageMaker (Level 300)
- Working with Scalable Machine Learning Algorithms in Amazon SageMaker
- Using Apache Spark with Amazon SageMaker
- Hyperparameter Tuning with Amazon SageMaker's Automatic Model Tuning
- Building Intelligent Applications with Machine Learning on AWS
- Machine Learning Models with TensorFlow Using Amazon SageMaker

Amazon SageMaker algorithms and model packages

As a seller of Amazon SageMaker products, you can list an algorithm, a model package, or both.

Amazon SageMaker algorithm

An Amazon SageMaker algorithm enables buyers to perform end-to-end machine learning. It has two logical components: training and inference. Buyers use the training component to create a training job in Amazon SageMaker and build a machine learning model. Amazon SageMaker saves the model artifacts generated by the algorithm during training to the buyer's Amazon Simple Storage Service (Amazon S3) bucket.

Buyers use the algorithm's inference component together with the model artifacts to build a model package which is then used to run real-time or batch transform jobs in Amazon SageMaker. As a seller, you can charge buyers for training and inference separately.
Amazon SageMaker model package

Model packages contain a pre-trained model that buyers can use to run real-time or batch inference jobs in Amazon SageMaker. They use the model for hosting services or running batch transforms in Amazon SageMaker. A model package contains an inference component that is packaged along with model artifacts that you provide. As a seller, you can build your model artifacts by training with Amazon SageMaker or you can use your own model artifacts from a previously built model. You can charge buyers for the inference jobs.

For more information on how to put your algorithms and models on the AWS Marketplace, see Putting your algorithms and model packages on the AWS Marketplace (p. 107).

Putting your algorithms and model packages on the AWS Marketplace

To distribute your products to AWS customers using AWS Marketplace, you must perform these steps.

Topics
- Package your code using Docker (p. 107)
- Create your algorithm in SageMaker (p. 108)
- Create your model package in SageMaker (p. 109)
- Add your algorithm or model package to AWS Marketplace (p. 110)
- Monetize your algorithm or model package (p. 111)

Package your code using Docker

SageMaker is a fully managed machine learning platform that provides flexibility for training and deploying models. Executable code – packaged using Docker containers – are run in secure and scalable infrastructure. Depending on your use case, SageMaker can be used in one of the following ways:

1. Use the SageMaker built-in algorithms to train and host models.
2. Write and use Python scripts to use in the machine learning frameworks like TensorFlow, PyTorch, MXNet, and Chainer.
3. Use Docker images with custom code.

SageMaker builds and maintains the containers for the first and second use cases. For the third use case, SageMaker lets customers package their own code into a Docker image. This code can be written in any programming language with any dependencies. These Docker images allow customers to run their code anywhere, without relying on the underlying hosting system.

Your Docker image size is governed by the Amazon ECR service limits in the Amazon Elastic Container Registry User Guide. The Docker image size affects the start-up time during training, batch transform, and endpoint creation jobs. For better performance, we recommend that you maintain the optimal Docker image size.

Package your algorithm code

Before packaging your image as a usable SageMaker product, we strongly recommend that you use SageMaker to test your custom training, batch transform, and real-time inference images. To package an algorithm or model on AWS Marketplace, you must provide a self-contained Docker image. By packaging
an algorithm in a container, you can use almost any code in SageMaker, regardless of programming language, environment, framework, or dependencies.

After packaging your code into a Docker image, create the algorithm by entering metadata that allows SageMaker to expose the algorithms to customers on the AWS Marketplace.

**Important**
When a buyer subscribes to your containerized product, the Docker containers are run in an isolated environment without internet. When you create your containers, do not rely on making outgoing calls over the internet as they will fail. Calls to AWS services will also fail.

**Package your inference code**
A model package can include the following elements:

- An inference container
- Optional model artifacts, which must be stored in Amazon S3

You create an inference container the same way as the algorithm container. Providing the location of the model artifacts in Amazon S3 is optional. You can choose to bundle your model artifacts within the inference container or let SageMaker retrieve them from your model storage location in Amazon S3.

**Create your algorithm in SageMaker**

An SageMaker algorithm has a training image and an inference image. You can use the same image to perform both training and inference, or you can choose to separate them. Training and inference images must remain compatible with each other and the model produced by the training image must be usable by the inference image. We recommend that you verify the validation outputs, specifically the batch transform output.

After packaging your code in Docker images, upload the images into Amazon Elastic Container Registry (Amazon ECR), and your Docker images will be scanned for known vulnerabilities. You also need to create an IAM role that allows SageMaker to access the image. For more information, see https://docs.aws.amazon.com/sagemaker/latest/dg/sagemaker-roles.html.

**To create your algorithm in SageMaker**

1. Open the SageMaker console and choose Create algorithm.
2. Choose your Amazon ECR image.
3. Enter the specifications for your algorithm.
4. Provide a training specification, supported instance types, hyperparameters, inference and channel information, the input data type, and validation specifications. You need a validation specification to sell the algorithm in AWS Marketplace. It is used to create training and inference jobs when validating the algorithm.
5. Your images will be scanned by default if you plan on selling your algorithm on AWS Marketplace.

Metadata helps buyers understand how to use your product and enables SageMaker to validate buyer requests synchronously after they have subscribed to your product.

**Validate your algorithm**

To ensure that buyers and sellers can be confident that products work in SageMaker, we require that you validate your algorithms before listing them on AWS Marketplace. To validate your algorithms, use your validation profile and sample data to run the following validation tasks:

1. Create a training job in your account to verify that your training image works with SageMaker.
2. Create a model in your account using the algorithm's inference image and the model artifacts produced by the training job.
3. Create a transform job in your account using the model to verify that your inference image works with SageMaker.

When you list your product on AWS Marketplace, the inputs and outputs of this validation process persist as part of your product and are made available to your buyers. This helps buyers understand and evaluate the product before they buy it. For example, buyers can inspect the input data that you used, the outputs generated, and the logs and metrics emitted by your code. The more comprehensive your validation specification, the easier it is for customers to evaluate your product.

To see the status of the jobs in your account, in the SageMaker console, see the Training jobs and Transform jobs pages in the console. If validation or scanning fail, you can access the scan and validation reports from the SageMaker console by clicking on their status. For additional details, you also can review the training and transform jobs that were created during validation. After fixing issues, recreate the algorithm. When the status of the algorithm is COMPLETED, find it in the SageMaker console and start the process of putting your product on the AWS Marketplace.

**Note**
Scanning and validation can take up to a few hours.

### Create your model package in SageMaker

After you have packaged your code as an inference container, and have the optional model artifacts stored in Amazon S3, you are ready to create the model package in SageMaker. To do this, use the following procedure:

**To create your model package**

1. Push the Docker image that you built to an Amazon ECR repository in your AWS account. For more information, see Pushing an Image in the Amazon Elastic Container Registry User Guide.
2. Set up permissions that allow SageMaker to access the Amazon ECR images and Amazon S3 objects.
3. Open the SageMaker console, choose **Create model package**, and follow the instructions.

To finish creating your model package, provide inference image and validation specifications, and your Docker images will be scanned for known vulnerabilities.

### Validate your model package

Before listing model packages on AWS Marketplace, you must validate them. This ensures that buyers and sellers can be confident that products work in Amazon SageMaker. You can list products on AWS Marketplace only if validation succeeds.

The validation procedure uses your validation profile and sample data to perform the following validations tasks:

1. Create a model in your account using the model package's inference image and the optional model artifacts that are stored in Amazon S3.
2. Create a transform job in your account using the model to verify that your inference image works with SageMaker.
3. Create a validation profile.

When you list your products on AWS Marketplace, the inputs and outputs of the validation process persist as part of your product and are made available to your buyers. This helps buyers understand and
evaluate the product before they buy it. For example, they can inspect the input data that you used, the outputs generated, and the logs and metrics emitted by your code. The more comprehensive the validation specification, the easier it is for customers to evaluate your product.

**Important**
In your validation profile, provide only data that you want to expose publicly.

To see the status of the jobs in your account, in the SageMaker console, see the **Training jobs** and **Transform jobs** pages in the console. If validation or scanning fail, you can access the scan and validation reports from the SageMaker console by clicking on their status. For additional details, you also can review the training and transform jobs that were created during validation. After fixing issues, recreate the algorithm. When the status of the algorithm is **COMPLETED**, find it in the SageMaker console and start the process of putting your product on the AWS Marketplace.

**Note**
Scanning and validation can take up to a few hours.

### Add your algorithm or model package to AWS Marketplace

After creating and validating your algorithm or model package in Amazon SageMaker, you can put your product on the AWS Marketplace. This process makes your products available in the AWS Marketplace and the SageMaker console.

**Note**
If you have not registered to sell on AWS Marketplace, review [Getting started as a seller (p. 3)](#) and complete the registration process.

After you've registered, do one of the following to add your product to AWS Marketplace:

- From the Amazon SageMaker console, choose the product, choose **Actions**, and choose **Publish new AWS Marketplace listing**. This carries over your product reference, the Amazon Resource Name (ARN), and directs you to the AMMP to create the listing.

- Navigate to **ML listing process**, manually enter the (ARN, and start listing your product. This process carries over the product metadata that you entered when you created the product in Amazon SageMaker. This information includes the training specification, supported instance types, hyperparameters, inference and channel information, input data type, and sample data.

When adding a product on AWS Marketplace, you'll provide the following:

- General product information
- Launch option
- Pricing and terms

**General product information**

Enter the product description, promotional resources, support information, and region availability. This information will appear on the AWS Marketplace product detail page. It is searchable in AWS Marketplace.

The resources for your product must include sample input data and sample notebooks that customers can use to get started with your model or algorithm. For more information, see [Best practices for sample input data and sample notebooks (p. 111)](#).

**Launch option**
Define general usage information, a customer-facing version number, and release notes. Review the Amazon SageMaker metadata, such as Amazon SageMaker content types, MIME types, supported input methods, and hyperparameters.

**Pricing and terms**

Define a EULA, pricing, the product tax code, and the refund policy. When you provide a paid algorithm, you can enter a training price for the algorithm and real-time and batch inference prices for the inference image packaged with the algorithm. When you list a model package, you can define real-time and batch inference prices for the package.

For both algorithms and model packages, you can define prices for each supported instance type per hour. You can enable a free-trial and specify the number of days for its duration.

**Product publishing**

You publish products through the AWS Marketplace Management Portal. The publishing process has a few steps, which allows you to review your product information one last time before it’s available for customers.

The first step requires that you provide general information, launch options, and set the pricing and terms for your product. The second step allows you to test your product before it’s live. At this point, your product has been published in a non-public limited state.

After testing has completed, you can choose **Sign off and publish** and select a version number for the product to be publicly available. At this point, your product is in a **Published (draft)** state.

Making your product public can take 30 to 60 minutes. If you try to access the AWS Marketplace product detail page during the publishing process, you might get a 404 error. This is expected while the information propagates through multiple systems.

**Monetize your algorithm or model package**

For algorithms and model packages, AWS Marketplace has an hourly pricing model per instance type. Algorithms have two prices: training price and inference price. Model packages have only an inference price. Amazon SageMaker supports real-time and batch inference modes, and you can set different prices for each mode. Buyer usage is metered and billed in one-second increments.

**Best practices for sample input data and sample notebooks**

It is important that developers, as well as machine learning practitioners, find it easy to try your models and algorithms. We strongly recommend that you provide the following information with your product.

1. Ideally **ten** input files, but at the very least one sample input file, attached under your product’s additional information section. If your model performs multi-class classification, you should provide at least one sample input file for each of the classes. This helps your customers understand the input format expected by the model/algorithm. Being able to see more sample input files helps users perform any necessary transformations on their data before performing inference, giving them the best results from your model.

2. At least **one** sample output file corresponding to one of the input files you provided, to be attached under your product’s additional information section. This helps your customers understand what kind of output to expect before having to go through the subscription process and improves the usability of your listing.
3. Provide the following information for your algorithms: the training data format, necessary pre-processing snippet, and specify both optional and mandatory features that can be provided by the user. Also specify whether the PIPE input mode is supported by the listing along with the input format required for it. Specify whether distributed training (more than 1 CPU/GPU instances) is supported. For tuning, provide the recommend hyperparameters.

4. From the usage information section of your models, provide a code snippet that shows data preparation steps and usage of the Invoke-endpoint (CLI/Python) API call to perform inference on the endpoint created from your model. This is crucial for your customers to know exactly how the payload should be sent.

**Example model usage information**
- Supported content types: image/jpeg, image/png, image/bmp
- Supported response types: application/json (default), image/jpeg
- Example CLI command:

  ```bash
  aws sagemaker-runtime invoke-endpoint --endpoint-name "endpoint_name" --body fileb:///img_name.jpeg --accept image/jpeg outfile.jpeg
  ```

If your models require data pre-processing, provide the necessary Python code snippet.

**Example data preprocessing information**

```python
import base64
image = open('image.jpeg', 'rb')
image_64_encode = base64.b64encode(image.read()).decode('utf-8')
#Prepare payload for prediction
payload="{"source": "+str(image_64_encode)+"\"\"")
```

5. Attach a sample notebook that demonstrates the end-to-end workflow under your product's additional information section. We recommend that you use the Pyhton SDK, instead of the Boto3 API in your sample notebook. Keep in mind that a well-developed sample notebook will make it easy for your customers to try and use your listing successfully. Keep the following sample notebook best practices in mind:

- For algorithms, your sample notebook should demonstrate end-to-end training, tuning, model creation, standing up endpoints, as well as how to perform inference and batch transform jobs on the model.
- For models, your sample notebook should demonstrate real-time inference, batch transform job inference, and provide a clear picture of the kind of data that is expected by the model.

For an example sample notebook that works in all regions, without entering any parameters or having to look for sample data, see `amazon_demo_product` on GitHub.

**Note**
A lack of training data would prevent your customers from running the notebook successfully. An under-developed sample notebook might prevent your customers from using it and hamper adoption.
Software as a service (SaaS)–based products

With software as a service (SaaS) products, you deploy software hosted on AWS infrastructure and grant buyers access to the software in your AWS environment. You are responsible for managing customer access, account creation, resource provisioning, and account management within your software.

For assistance with your SaaS products, contact us.

Topics
- Getting started (p. 113)
- Plan your SaaS product (p. 119)
- SaaS product guidelines (p. 121)
- Pricing SaaS products (p. 122)
- SaaS customer onboarding (p. 126)
- Amazon SNS notifications for SaaS products (p. 128)
- Accessing the AWS Marketplace Metering and Entitlement Service APIs (p. 129)
- Reporting (p. 134)
- Code examples (p. 134)
- Using AWS PrivateLink with AWS Marketplace (p. 137)

Getting started

This chapter outlines how SaaS products work for sellers who create and maintain them. This section describes how to get your SaaS product on AWS Marketplace and how to integrate it with the appropriate AWS Marketplace APIs, based on the SaaS product's billing model.

Prerequisites

Before you get started, you must complete the following prerequisites:

1. Access and use the AWS Marketplace Management Portal. This is the tool that you use to register as a seller and manage the products that you sell on AWS Marketplace.
2. Register as a seller and submit your tax and banking information. For more information, see Seller registration process (p. 5).
3. Plan how you'll create and integrate your SaaS product in AWS Marketplace. For more information, see Plan your SaaS product (p. 119).

To get started, choose one of the following:
- SaaS subscriptions (p. 114)
- SaaS contracts (p. 116)
- SaaS contracts with consumption (p. 117)
**SaaS subscriptions**

To list and maintain a SaaS product with the subscription pricing model on AWS Marketplace, follow the procedures in this topic.

Before you begin, make sure you've chosen the right pricing model for your SaaS product in AWS Marketplace. For more information, see Plan your SaaS product (p. 119).

**Procedures**

- List your SaaS subscription product on AWS Marketplace (p. 114)
- Conduct AWS integration testing (p. 115)
- Review your SaaS AWS Marketplace product page before going live (p. 115)

**List your SaaS subscription product on AWS Marketplace**

The following process outlines the steps you must take to list your SaaS subscription product on AWS Marketplace.

**Gather your product information**

Before you create your product on AWS Marketplace, gather the following information:

- A SaaS application that you can list as a product with a subscription billing model on AWS Marketplace.
- Product logo URL. A publicly accessible URL that contains a clear image of the logo for the product you're providing.
- Your product's End User License Agreement (EULA) URL. Your product must have an EULA and you must provide a link to it for customers to read and review on your product's AWS Marketplace page.
- Your product's registration URL. This is where customers will be sent after they subscribe to your product on AWS Marketplace.
- Metadata about your product, as defined in the product creation wizard of the AWS Marketplace Management Portal.
- Support information for your product. This includes email addresses and URLs for your product's support channels.

**Create a SaaS product**

Take your SaaS application information and create a new SaaS product in the AWS Marketplace Management Portal.

2. For **Products**, choose **SaaS**.
3. For **Create SaaS product**, choose **SaaS Subscriptions**, and then choose **Start**.
4. Read through and fill out the product creation wizard using the information you gathered. For assistance with creating your SaaS subscription product, contact us.
5. The AWS Marketplace Operations team publishes your product as a limited product stage that is visible to you and any AWS accounts you have allowed to view the product.

**Note**

Prices can be temporarily reduced to enable you to test the purchase flow without incurring high charges. For more information, contact us.
6. The AWS Marketplace Operations team sends an email to the address associated with your AWS account to enable testing of product codes, Amazon SNS topics, and product page URLs. This is the first of several tests for your product that are required before the product can go live.

**Conduct AWS integration testing**

After you've created the product, you must conduct in-depth AWS integration testing.

1. Use an allowed account to test the customer experience by subscribing to your product.
2. After you've subscribed with the allowed account, ensure that the account is redirected to the registration URL, and that the redirect is a POST request that includes a temporary token. Then, your SaaS application must do the following:
   - Exchange the token for a `customerID` by calling the `ResolveCustomer` action in the AWS Marketplace Metering Service.
   - Persist the `customerID` in your application for future calls.
3. After verifying the test account in the previous step, onboard the account into your application. For example, you can have the test customer fill out a form to create a new user account. Or, provide them with other next steps to get access to your SaaS application.
4. After they're onboarded, send metering records to AWS for billing purposes using the `BatchMeterUsage` action in the AWS Marketplace Metering Service. We recommend using AWS CloudTrail to monitor activity to ensure that billing information is being sent to AWS. Keep the following in mind when sending metering records:
   - Metering requests are de-duplicated on the hour.
   - Records sent every hour are cumulative.
   - Even if there were no records in the last hour, we strongly recommend as a best practice that you send metering send records every hour.
5. Test for subscription changes by setting up an Amazon SQS queue and subscribing to your product's Amazon SNS topic. The Amazon SNS topic provides notifications about changes to customer subscriptions. This enables you to know when to provide and revoke access for specific customers. Possible scenarios include unsubscribes, successful subscriptions, and failed subscriptions.
6. Verify a successful subscription. After you receive an Amazon SNS notification for your test account with a successful subscription message, metering can begin. Records that are sent to the AWS Marketplace Metering Service before you receive the Amazon SNS notification aren't metered.
   
   **Note**
   To prevent billing issues, we strongly recommend programmatically waiting for this notification before launching resources on behalf of your customers.
7. After you have completed all of the integration requirements and tested the solution, notify the AWS Marketplace Operations team. They will run a series of final tests on the solution by verifying that you have successfully sent metered records with the `BatchMeterUsage` action.

For additional information, see **Metering for usage** (p. 130).

**Review your SaaS AWS Marketplace product page before going live**

After end-to-end testing is complete, you need to review the product page with the original prices. After you approve the page, the AWS Marketplace Operations team will make the product page live on AWS Marketplace. At this point, customers can start discovering and subscribing to your product.
SaaS contracts

To list and maintain a SaaS product with the contract pricing model on AWS Marketplace, follow the procedures in this topic.

Before you begin, make sure you've chosen the right pricing model for your SaaS product in AWS Marketplace. For more information, see Plan your SaaS product (p. 119).

Procedures

- List your SaaS contract product on AWS Marketplace (p. 116)
- Conduct AWS integration testing (p. 117)
- Review your SaaS AWS Marketplace product page before going live (p. 117)

List your SaaS contract product on AWS Marketplace

The following process outlines the steps you must take to list your SaaS contract product on AWS Marketplace.

Gather your product information

Before you create your product on AWS Marketplace, gather the following information:

- A SaaS application that you can list as a product with a contract billing model on AWS Marketplace.
- Product logo URL. A publicly accessible URL that contains a clear image of the logo for the product you're providing.
- Your product's End User License Agreement (EULA) URL. Your product must have an EULA and you must provide a link to it for customers to read and review on your product's AWS Marketplace page.
- Your product's registration URL. This is where customers will be sent after they subscribe to your product on AWS Marketplace.
- Metadata about your product, as defined in the product creation wizard of the AWS Marketplace Management Portal.
- Support information for your product. This includes email addresses and URLs for your product's support channels.

Create a SaaS product

Take your SaaS application information and create a new SaaS product in the AWS Marketplace Management Portal.

2. For Products, choose SaaS.
3. For Create SaaS product, choose SaaS Contracts, and then choose Start.
4. Read through and fill out the product creation wizard using the information you gathered previously. For assistance with creating your SaaS contract product, contact us.
5. The AWS Marketplace Operations team will publish your product as a limited product stage visible to you and any AWS accounts you have allowed to view the product.

Note
Prices can be temporarily reduced to enable you to test the purchase flow without incurring high charges, contact us for more information.
6. The AWS MP Ops team will send an email to the address associated with your AWS account to enable testing of product codes, Amazon SNS topics, and product page URLs. This is the first of several tests for your product that are required before the product can go live.

**Conduct AWS integration testing**

Once the product has been created, more in-depth testing can begin. The following test must be completed:

1. Use an allowed account to test the customer experience by getting a contract for your product.
2. Once the account has the contract, ensure that the account is redirected to the registration URL, and that the redirect is a POST request that includes a temporary token. Then your SaaS application must:
   - Exchange the token for a customerID by calling the `ResolveCustomer` action in the AWS Marketplace Metering Service.
   - Persist the customerID in your application for future calls.
   - With the customerID, call `GetEntitlement` in the AWS Marketplace Entitlement Service to verify which dimension the customer is subscribed to and the quantity, if applicable.
3. After verifying the test account in the previous step, onboard the account into your application. For example, you can have the test customer fill out a form to create a new user account. Or provide them with other next steps to get access to your SaaS application.
4. If no entitlement is returned from `GetEntitlement`, either during onboarding or in your ongoing verification passes, determine how to manage access and the experience for unentitled users.
5. Test for subscription changes by setting up an Amazon SQS queue and subscribing to your product's Amazon SNS topic. The Amazon SNS topic provides notifications about changes to customers subscription. This enables you to know when to provide and revoke access for specific customers. Possible scenarios include unsubscribes, successful subscription, and failed subscription.
6. After you have completed all the integration requirements and tested the solution, notify the AWS Marketplace Ops team. They will then test the solution by verifying you have successfully called `GetEntitlement` and sufficiently onboard new customers. They will also verify you have successfully sent metered records via `BatchMeterUsage`.

For additional information, see Checking entitlements (p. 131).

**Review your SaaS AWS Marketplace product page before going live**

After end-to-end testing is complete, you will have the chance to review the product page with the original prices. After giving approval, the AWS Marketplace Ops team will make the product page live on AWS Marketplace. At this point, customers can start discovering and subscribing to your product.

**SaaS contracts with consumption**

To list and maintain a SaaS product with the contract with consumption pricing model on AWS Marketplace follow the procedures in this topic.

Before you begin, make sure you've chosen the right pricing model for your SaaS product in AWS Marketplace. For more information, see Plan your SaaS product (p. 119).

**Procedures**

- List your SaaS contract product on AWS Marketplace (p. 118)
• Conduct AWS integration testing (p. 118)
• Review your SaaS AWS Marketplace product page before going live (p. 119)

List your SaaS contract product on AWS Marketplace

The following process outlines the steps you must take to list your SaaS contract with consumption product on AWS Marketplace:

Gather your product information

Before you create your product on AWS Marketplace, gather the following information:

• A SaaS application that you can list as a product with a contract with consumption billing model on AWS Marketplace.
• Product logo URL. A publicly accessible URL that contains a clear image of the logo for the product you’re providing.
• Your product’s End User License Agreement (EULA) URL. Your product must have an EULA and you must provide a link to it for customers to read and review on your product’s AWS Marketplace page.
• Your product’s registration URL. This is where customers will be sent after they subscribe to your product on AWS Marketplace.
• Metadata about your product, as defined in the product creation wizard of the AWS Marketplace Management Portal.
• Support information for your product. This includes email addresses and URLs for your product’s support channels.

Create a SaaS product

Take your SaaS application information and create a new SaaS product in the AWS Marketplace Management Portal.

2. From the Products dropdown menu, choose SaaS.
3. From Create SaaS product, choose SaaS Contracts, and then choose Start.
4. Read through and fill out the product creation wizard using the information you gathered previously. For assistance with creating your SaaS contract product, contact us.
5. The AWS MP Ops team will publish your product as a limited product stage visible to you and any AWS accounts you have allowed to view the product.

   **Note**
   Prices can be temporarily reduced to enable you to test the purchase flow without incurring high charges, contact us for more information.

6. The AWS MP Ops team will send an email to the address associated with your AWS account to enable testing of product codes, Amazon SNS topics, and product page URLs. This is the first of several tests for your product that are required before the product can go live.

Conduct AWS integration testing

Once the product has been created, more in-depth testing can begin. The following test must be completed:

1. Use an allowed account to test the customer experience by getting a contract for your product.
Plan your SaaS product

2. Once the account has the contract, ensure that the account is redirected to the registration URL, and that the redirect is a POST request that includes a temporary token. Then your SaaS application must:

- Exchange the token for a customerID by calling the ResolveCustomer action in the AWS Marketplace Metering Service.
- Persist the customerID in your application for future calls.
- With the customerID, call GetEntitlement in the AWS Marketplace Entitlement Service to verify which dimension the customer is subscribed to and the quantity, if applicable.

3. After verifying the test account in the previous step, onboard the account into your application. For example, you can have the test customer fill out a form to create a new user account. Or provide them with other next steps to get access to your SaaS application.

4. If no entitlement is returned from GetEntitlement, either during onboarding or in your ongoing verification passes, determine how to manage access and the experience for unentitled users.

5. Once onboarded, send metering records to AWS for billing purposes using the BatchMeterUsage action in the AWS Marketplace Metering Service. We recommend using AWS CloudTrail to monitor activity to ensure that billing information is being sent to AWS. Keep the following in mind when sending metering records:

- Metering requests are de-duplicated on the hour.
- Records sent every hour are cumulative.
- Even if there were no records in the last hour, we strongly recommend as a best practice that you send metering send records every hour.

6. Setup an Amazon SQS queue and subscribe to your products Amazon SNS topics. These topics provide notifications about changes to customers subscription and entitlement statuses. This enables you to know when to provide and revoke access for specific customers. Possible scenarios include: unsubscribes, upgrades, renewals, and failed subscription.

7. After you have completed all the integration requirements and tested the solution, notify the AWS Marketplace Ops team. They will then test the solution by verifying you have successfully called GetEntitlement and sufficiently onboard new customers. They will also verify you have successfully sent metered records via BatchMeterUsage.

For additional information, see Checking entitlements (p. 131).

Review your SaaS AWS Marketplace product page before going live

After end-to-end testing is complete, you will have the chance to review the product page with the original prices. After giving approval, the AWS Marketplace Ops team will make the product page live on AWS Marketplace. At this point, customers can start discovering and subscribing to your product.

Plan your SaaS product

Before you add your SaaS product to AWS Marketplace, you must first do some planning. This step is critical to the success of your product. A lack of planning can result in billing issues or you might have to re-create your product in AWS Marketplace.

Important
Most of your product's settings can't be changed after you've configured them. If you need to change them after the product is created in AWS Marketplace, you probably need to create a new product with the correct settings.
Plan your pricing

There are three pricing options for SaaS products on AWS Marketplace. Choosing the right pricing model for your product is the most important decision you will make. Choosing the wrong pricing model can set you back by weeks, because it determines the payment options for your customers and the billing integration code you'll need to write, test, and deploy.

- **SaaS subscriptions** – A pay-as-you-go model where buyers are billed for their hourly usage of your SaaS product.
- **SaaS contracts** – Buyers are either billed in advance for the use of your software, or you can offer them a flexible payment schedule.
- **SaaS contracts with pay-as-you-go** – This option is similar to a standard contract, however your customers can also pay for additional usage above their contract. This is a blended pricing option, that gives your customers the most pricing options and it requires the most integration code on your end.

For more information on pricing, see Pricing SaaS products (p. 122).

Plan your billing integration

One of the benefits of having a SaaS product on AWS Marketplace is consolidating billing. In order to take advantage of this benefit, you must integrate with the AWS Marketplace Metering Service or the AWS Marketplace Entitlement Service, depending on your chosen pricing model. These two services help you ensure that your billing and usage reporting is accurate.

After you plan your integration, you must test the integration with your product before it goes live. For more information about integration and testing, see Accessing the AWS Marketplace Metering and Entitlement Service APIs (p. 129).

Plan your Amazon SNS integration

There are two Amazon SNS topics that you can subscribe to for your SaaS product. These messages can help you programmatically handle changes to subscriptions and contracts initiated by AWS or by your customers. You can use these Amazon SNS notifications as programmatic triggers to enable a customer to register for a new account in on your product registration website, to deny customers with expired subscriptions from accessing your product, depending on how you program the handling of these notifications.

Plan how customers will access your product

This section describes how to make your product accessible to buyers.

Plan your SaaS product registration Website

Customers who buy your SaaS product need to access to it. You must plan and implement how you want your customers to access the product. SaaS products support the following access options:

- AWS PrivateLink
- Your own product website

Using AWS PrivateLink for customers to access your SaaS product

You can use Using AWS PrivateLink with AWS Marketplace (p. 137) to configure your service as an Amazon Virtual Private Cloud (Amazon VPC) endpoint service. Your customers can create a VPC endpoint and access your software across the AWS Cloud virtual network. Alternatively, you can provide access to
your software product through a website you own and maintain, with customers creating a connection across the internet.

**Using your own registration website**

Your SaaS product is hosted in your environment and it must be accessed over the internet through a public endpoint that you manage and maintain, like a website. Typically, you have a website that customers use to register for your product, sign in to use the product, and access support for your product. For the sake of simplicity, this endpoint will be referred to as your registration website.

If you choose this access option and your product doesn't already have a registration website, you need to create one. After you have a registration website, your website must be programmed to validated customers whenever they access your registration page.

**To validate customers using your registration website**

1. Accept POST requests that includes the temporary token `x-amzn-marketplace-token`.
2. Exchange the token for a `customerID` by calling `ResolveCustomer` in the AWS Marketplace Metering Service.
3. After obtaining a `customerID`, persist it in your application for future calls.
4. With the `customerID`, call `GetEntitlement` in the AWS Marketplace Entitlement Service to verify which dimension the customer is subscribed to and the quantity.
5. After you've verified your customer's access and entitlement, program your application to ensure that the customer doesn't exceed what they're entitled to.

**SaaS product guidelines**

AWS Marketplace maintains these guidelines for all SaaS products and offerings on AWS Marketplace to promote a safe, secure, and trustworthy platform for our customers.

All products and their related metadata are reviewed when submitted to ensure that they meet or exceed current AWS Marketplace guidelines. These guidelines are reviewed and adjusted to meet our evolving security requirements. In addition, AWS Marketplace continuously reviews products to verify that they meet any changes to these guidelines. If products fall out of compliance, we might require that you update your product and in some cases your product might temporarily be unavailable to new subscribers until issues are resolved.

**Product setup guidelines**

All SaaS products must adhere to the following product setup guidelines:

- At least one pricing dimension must have a price greater than $0.00.
- All pricing dimensions must relate to actual software and cannot include any other products or services unrelated to the software.
- SaaS products offered exclusively in the AWS GovCloud (US) Regions must include `GovCloud` somewhere in the product title.

**Customer information requirements**

All SaaS products must adhere to the following customer information requirements:

- SaaS products must be billed entirely through the listed dimensions on AWS Marketplace.
- You cannot collect customer payment information for your SaaS product at any time, including credit card and bank account information.
Product usage guidelines

All SaaS products must adhere to the following product usage guidelines:

• After subscribing to the product in AWS Marketplace, customers should be able to create an account within your SaaS application and gain access to a web console within two business days. If the customer cannot gain access to the application immediately, you must provide a message with specific instructions on when they will gain access. When an account has been created, the customer must be sent a notification confirming that their account has been created along with clear next steps.

• If a customer already has an account in the SaaS application, they must have the ability to log in from the fulfillment landing page.

• Customers must be able to see the status of their subscription within the SaaS application, including any relevant contract or subscription usage information.

• Customers must be able to easily get help with issues such as: using the application, troubleshooting, and requesting refunds (if applicable). Support contact options must be specified on the fulfillment landing page.

Architecture guidelines

All SaaS products must adhere to the following architecture guidelines:

• A portion of your application must be hosted in an AWS account that you own.

• All application components should be hosted in infrastructure you manage. Applications that require additional resources in the customer’s infrastructure must follow these guidelines:
  • Provision resources in a secure way, such as using the AWS Security Token Service (AWS STS) or AWS Identity and Access Management (IAM).
  • Provide additional documentation including a description of all provisioned AWS services, IAM policy statements, and how an IAM role or user is deployed and used in the customer’s account.
  • Include a notification in the product description that explains that if the customer incurs additional AWS infrastructure charges separate from their AWS Marketplace transaction, they’re responsible for paying the additional infrastructure charges.
  • If your product deploys an agent, you must provide instructions to the customer that describe how to deploy it in their AWS account.

• Successfully call the AWS Marketplace APIs from the AWS account that registered as a provider and submitted the SaaS publishing request. The SaaS pricing model determines which APIs should be called:
  • SaaS contracts – GetEntitlements in the AWS Marketplace Entitlement Service.
  • SaaS contracts with consumption – GetEntitlements in the AWS Marketplace Entitlement Service and BatchMeterUsage in the AWS Marketplace Metering Service.
  • SaaS subscriptions – BatchMeterUsage in the AWS Marketplace Metering Service.
  • SaaS products offered exclusively in the AWS GovCloud (US) Regions must outline the architectural boundaries between other AWS Regions and the AWS GovCloud (US) Regions, use cases for the product, and the workloads not recommended for the product.

Pricing SaaS products

After a buyer gets your SaaS product on AWS Marketplace, AWS Marketplace passes along their billing identifier. You use the billing identifier to call the AWS Marketplace Entitlement Service and the AWS Marketplace Metering Service. Then customers access the product in your AWS environment or through a VPC endpoint connection you create. AWS Marketplace offers the following pricing models for SaaS products:
• **SaaS subscriptions** – A pay-as-you-go model where we bill buyers for their hourly usage of your SaaS product.

• **SaaS contracts** – Buyers are either billed in advance for the use of your software, or you can offer them a flexible payment schedule. Customers can also pay for additional usage above their contract.

To make your SaaS product available on AWS Marketplace, decide whether you want to offer the SaaS subscriptions pricing model or the SaaS contracts pricing model.

### Topics
- Pricing for SaaS subscriptions (p. 123)
- Pricing for SaaS contracts (p. 124)

### Pricing for SaaS subscriptions

For SaaS subscriptions, AWS Marketplace bills your customers based on the metering records that you send to us. Before you can publish a subscription-based SaaS product you must do the following:

2. Complete the wizard with the necessary information.

To set your pricing, select the category that best describes your product’s pricing. The pricing category appears to customers on the AWS Marketplace website. You can choose from bandwidth (GBps, MBps), data (GB, MB, TB), hosts, requests, tiers, or users. If none of the predefined categories fit your needs, you can choose the more generic **units** category.

Next, define your pricing dimensions. Each pricing dimension represents a feature or service that you can set a per-unit price for. Examples of dimensions include users, hosts scanned, and GB of logs ingested. You can define up to 24 dimensions. For each dimension you define, you must add the following information:

- **Dimension API Name** – The API name used when sending metering records to the [AWS Marketplace Metering Service](https://aws.amazon.com/marketplace-metering-service). This name indicates which dimension your customer used. This name is visible in billing reports. The name doesn't need to be reader-friendly because you're the only one with access to your reports. After you set the name, you can't change it.

- **Dimension Description** – The customer-facing statement that describes the dimension for the product. The description (administrators per hour, per Mbps bandwidth provisioned, etc.) can be no more than 70 characters and should be user-friendly. After the product is published, you can't change this description.

- **Dimension Price** – The software charge per unit for this product, in USD. This field supports three decimal places.

### When a SaaS subscription ends

A customer can unsubscribe from your SaaS subscription product through the AWS Management Console.

1. Your SaaS product is sent an unsubscribe-pending notification through the Amazon SNS topic for that customer.
2. You have one hour to meter any remaining usage for the customer.
3. After this hour, you receive an unsubscribe-success notification. At this point, you can no longer send metering records for this customer.
It's up to you to decide how you want to disable functionality in your SaaS product for unsubscribed customers. For example, your product might complete the customer's existing work, but prevent them from creating work. You might want to display a message to the customer that their usage has been disabled. Customers can resubscribe to your product through AWS Marketplace.

Subscription Cancellations

Customers cancel SaaS subscription products through the **Your Marketplace Software** page of the AWS Marketplace website. When a customer cancels a subscription, you receive a notification, and you have 1 hour to send a final metering record for the customer. You notify the customer from your product that the cancellation is in progress. If a customer indicates that they want to cancel through your product, direct the customer to AWS Marketplace. To guarantee that there will be no future charges, customers should confirm the cancellation with AWS Marketplace.

Customers can request a cancellation and refund for SaaS contract products through AWS Support. Customers must request refunds within 48 hours through AWS Support. The full or prorated refund is typically granted in 3–5 business days. When a customer cancels a contract, you receive a notification, and you have 1 hour to send a final metering record for the customer for any additional usage charges.

Pricing for SaaS contracts

For SaaS contracts, AWS Marketplace bills your customers upfront or by the payment schedule that you define, based on the contract between you and your customer. After that point, they're entitled to use those resources. For additional usage above their contract, AWS Marketplace bills your customers based on the metering records received by us through the AWS Marketplace Metering Service.

2. Complete the wizard with the necessary information.

To set your pricing, choose one or more contract durations you offer customers. You can enter different prices for each contract duration. Your options are monthly, 1-year, 2-year, and 3-year durations.

Choose the category that best describes your product’s pricing. The pricing category appears to customers on the AWS Marketplace website. You can choose from bandwidth (GB/s, MB/s), data (GB, MB, TB), hosts, requests, tiers, or users. If none of the predefined categories fit your needs, you can choose the more generic **units** category.

For **Enable Tiered Dimensions**, choose how you want your customers to be able to purchase your product from the following options:

- **Buyer can choose only one tier offered** – Customers choose a tier from options that include different sets of features, services, and usage amounts.
- **Buyer can choose one or more options offered** – Customers can select a quantity for each pricing dimension you offer.

After you choose a category, define your pricing dimensions. Each pricing dimension represents a feature or service that you can set a per unit price for. Examples of dimensions are users, hosts scanned, and GB of logs ingested. For each dimension you define, you add a name, a description, a price, and an API name. The name, price, and description are displayed to customers. You use the API name for tracking and reporting with AWS Marketplace as follows:

- When calling the **AWS Marketplace Entitlement Service** to retrieve the dimensions your customers have purchased.
- When calling the **AWS Marketplace Metering Service** to indicate which dimensions customers used.
For each pricing dimension that you add to your contract, you can choose to let customers pay as they go for additional usage of that dimension above their contract. You can also add additional dimensions without contract prices that customers only consume by paying as they go.

When using the wizard to create the contracts for your SaaS product, you must define the following fields for your pricing dimensions:

- **Dimension API Name** – The name used when calling the Entitlements API. This name is visible in billing reports and reports are not external-facing. The maximum length for the API name is 15 characters, and after you set the name it can’t be changed.

- **Dimension Display Name** – The customer-facing name of a dimension. This name should help the customer understand the dimension for the product. The name should be user-friendly and the maximum length is 24 characters. This value can be changed.

- **Dimension Description** – The customer-facing description of a dimension that provides additional information about the dimension for the product. The maximum length for the description is 70 characters.

- **Dimension - Monthly Price** – The software charge per unit for the 1-month option for this dimension. This field supports three decimal places.

- **Dimension - 1 Year Price** – The software charge per unit for the 12-month option for this dimension. This field supports three decimal places. It’s not a monthly charge. The price must reflect the 12-month one-time charge price.

- **Dimension - 2 Years Price** – The software charge per unit for the 24-month option for this dimension. This field supports three decimal places.

- **Dimension - 3 Years Price** – The software charge per unit for the 36-month option for this dimension. This field supports three decimal places.

**Example: Data storage application**

<table>
<thead>
<tr>
<th></th>
<th>Monthly price</th>
<th>12-month price</th>
<th>24-month price</th>
<th>Pay-as-you-go price for additional usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unencrypted data (GB)</td>
<td>$1.50/GB</td>
<td>$16.00/GB</td>
<td>$30.00/GB</td>
<td>$0.1/GB per hour</td>
</tr>
<tr>
<td>Encrypted data (GB)</td>
<td>$1.55/GB</td>
<td>$16.60/GB</td>
<td>$31.20/GB</td>
<td>$0.11/GB per hour</td>
</tr>
</tbody>
</table>

**Example: Log monitoring product**

<table>
<thead>
<tr>
<th></th>
<th>Monthly price</th>
<th>12-month price</th>
<th>Pay-as-you-go price for additional usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic (10 hosts monitored, 5 containers monitored)</td>
<td>$100</td>
<td>$1000</td>
<td></td>
</tr>
<tr>
<td>Standard (20 hosts monitored, 10 containers monitored)</td>
<td>$200</td>
<td>$2000</td>
<td></td>
</tr>
<tr>
<td>Pro (40 hosts monitored, 20 containers monitored)</td>
<td>$400</td>
<td>$4000</td>
<td></td>
</tr>
</tbody>
</table>
### AWS Marketplace Seller Guide

#### SaaS customer onboarding

<table>
<thead>
<tr>
<th></th>
<th>Monthly price</th>
<th>12-month price</th>
<th>Pay-as-you-go price for additional usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional hosts</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>monitored per hour</td>
<td></td>
<td></td>
<td>$0.1</td>
</tr>
<tr>
<td>Additional containers</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>monitored per hour</td>
<td></td>
<td></td>
<td>$0.2</td>
</tr>
</tbody>
</table>

**Note**

The prices can be for the following durations: 1 month, 12 months, 24 months, or 36 months. You can choose to offer one or more of these options for your product. The durations must be the same across each dimension. For example, in a case where you have `ReadOnlyUsers` and `AdminUsers` dimensions, if you offer a yearly price for `ReadOnlyUsers`, you must offer a yearly price for `AdminUsers`, too.

### Upgrades

Customers can upgrade a contract to one of a higher value except for longer durations. For example, they can upgrade to higher quantities or higher-value entitlements. Customers are given a prorated credit for their existing contract. Customers can't decrease the size of their existing contract. They can only decrease the size at renewal, or cancel their renewal.

Entitlements are verified by your SaaS product, which makes calls to the AWS Marketplace Entitlement Service.

### Automatic renewals

When a customer purchases your product through AWS Marketplace using SaaS contracts, they can agree to automatic renewal of the contract terms. The customer continues to pay for the entitlements every month or for 1, 2, or 3 years. The customer always has the option to modify the renewal settings. They can cancel the renewal or renew the contract different quantities and durations.

### When a SaaS contract ends

A SaaS contract product has a contract expiry. When a contract ends, the following events occur:

1. Your SaaS product receives an `entitlement-updated` notification indicating their entitlement has changed, and the AWS Marketplace Entitlement Service returns an empty response.
2. You have one hour to meter any remaining usage for the customer.
3. After this you can no longer send metering records for this customer.

### SaaS customer onboarding

With SaaS subscriptions and SaaS contracts, your customers subscribe to your products through AWS Marketplace, but access the product in your AWS environment. After subscribing to the product, your customer is directed to a website you create and manage as a part of your SaaS product to register their account and configure the product.

When creating your product, you provide a URL to your registration landing page. We use that URL to redirect customers to your registration landing page after they subscribe. On your software's registration URL, you collect whatever information is required to create an account for the customer. We recommend
collecting your customer’s email addresses if you plan to contact them through email for usage notifications.

The registration landing page needs to be able to identify and accept the x-amzn-marketplace-token token in the form data from AWS Marketplace with the customer’s identifier for billing. It should then pass that token value to the AWS Marketplace Metering Service and AWS Marketplace Entitlement Service APIs to resolve for the unique customer identifier and corresponding product code. For a code example, see ResolveCustomer code example (p. 135).

Configuring your SaaS product to accept new buyers

You’re responsible for correctly configuring your SaaS software to accept new customers and meter them appropriately. The following process outlines one recommended way of identifying, implementing, and metering a new customer’s access to your software:

1. When a customer visits your product page on the AWS Marketplace website, they choose to subscribe to your product.
2. The customer’s AWS account is subscribed to your product. This means metering records sent from your product become part of the customer’s AWS bill.
3. A registration token is generated for the customer that contains their customer identifier to your website.
4. The customer is redirected to your software’s registration URL. This page must be able to accept the token with the customer’s identifier.
5. The customer’s browser sends a POST request to your SaaS registration URL. The request contains one POST parameter, x-amzn-marketplace-token, containing the customer’s registration token. From the perspective of your registration website, the customer has submitted a form with this parameter. The registration token is an opaque string.
6. To redeem this token for a customer identifier and a product code, your website must call ResolveCustomer on the AWS Marketplace Metering Service. The customer identifier isn’t the customer’s AWS account ID, but it’s universal between products. The product code is a unique string for your SaaS product that AWS provides to you. Each AWS product has one unique product code, which is assigned to you during registration.

The following is an example of a ResolveCustomer call.

```
##### Resolving Customer Registration Token #####
formFields = urlparse.parse_qs(postBody):
if formFields.has_key('x-amzn-marketplace-token'):
    marketplaceClient = boto3.client('meteringmarketplace')
    customerData = marketplaceClient.resolve_customer(
        RegistrationToken=formFields['x-amzn-marketplace-token'],
        productCode = customerData['ProductCode'],
        customerId = customerData['CustomerIdentifier'])
    # TODO: Store information away with your customer record
    # TODO: Validate no other accounts share this identifier
```

7. Your website validates that the product code matches your SaaS product identity. Your website must keep this customer identifier in the customer’s session. It can be stored temporarily on your server, or it can be part of a signed session cookie on the customer’s browser.
8. The customer is instructed to either create an account in your product or sign in to an existing account.
9. The customer is now signed in to your website using credentials specific to that SaaS product. In your accounts database, you can have a row for each customer. Your accounts database must have a column for the AWS customer identifier, which you populate with the customer identifier that
you obtained in step 2. Verify that no other accounts in your system share this customer identifier. Otherwise, you might send conflicting metering records.

10. During your seller registration process, you are assigned an Amazon SNS topic that notifies you when customers subscribe or unsubscribe to your product. The notification is an Amazon SNS notification in JSON format that informs you of customer actions.

We recommend that you use Amazon Simple Queue Service (Amazon SQS) to capture these messages. After you receive a subscription notification with `subscribe-success`, the customer account is ready for metering. Records that you send before this notification aren't metered. For information about how to do this, see Step 2: Give Permission to the Amazon SNS Topic to Send Messages to the Amazon SQS Queue in the Amazon Simple Notification Service Developer Guide.

If you have a SaaS contracts product, you also get an `entitlement-updated` notification when the contract is created. Your accounts database must have an extra column for the subscription state. The following is an example of a `subscribe-success` subscription notification.

```json
{
  "action": "subscribe-success",
  "customer-identifier": "T1EXAMPLEjMONTIzNDEyMaQtNTY3ODU2ODc1EXAMPLEn",
  "product-code": "72EXAMPLE2dg8dfEXAMPLEn"
}
```

**Note**
Do not activate a product subscription unless you receive a `SUBSCRIPTION_SUCCESSFUL` notification.

11. Use the customer identifier stored in your database to meter for usage through the AWS Marketplace Metering Service or check for entitlements through the AWS Marketplace Entitlement Service.

**Security and ordering**

As a seller, it's your responsibility to trust only customer identifiers that are immediately returned from AWS or those that your system has signed. We recommend that you resolve the registration token immediately because it expires after 1 hour. After you resolve the registration token, store the customer identifier as a signed attribute on the customer's browser session until the registration is complete.

**Amazon SNS notifications for SaaS products**

To receive notifications, you subscribe to the AWS Marketplace Amazon Simple Notification Service (Amazon SNS) topic provided to you during product creation. The topic provides notifications about changes to customers' subscription and contract entitlement statuses. This enables you to know when to provide and revoke access for specific customers.

The following Amazon SNS topics are specific to software as a service (SaaS) products:

- `aws-mp-entitlement-notification` – This Amazon SNS topic is for SaaS contracts.
- `aws-mp-subscription-notification` – This Amazon SNS topic is for SaaS subscriptions and contracts with additional consumption.

**Note**
If your product is priced for SaaS contracts with consumption, you must use both of these topics.
SaaS product Amazon SNS message body

Each message for the SaaS product Amazon SNS notifications has the following format.

{
  "action": "action-name",
  "CustomerIdentifier": "X01EXAMPLEX",
  "ProductCode": "n0123EXAMPLEXXXXXXXXXXXXX"
}

SaaS product Amazon SNS actions

As a SaaS contract provider, you'll receive messages with the entitlement-updated action. When you get one of these messages, a subsequent call to the GetEntitlement AWS Marketplace Entitlement Service action is required to discover the content of the update.

If you provide a SaaS subscription product (or a SaaS contract with consumption product), you'll receive messages with the following actions:

- subscribe-success
- subscribe-fail
- unsubscribe-pending
- unsubscribe-success

Subscribing an SQS queue to the SNS topic

We recommend subscribing an Amazon SQS queue to the provided SNS topic. For detailed instructions on creating an SQS queue and subscribing the queue to the provided topic, see Using Amazon SNS for system-to-system messaging with an Amazon SQS queue as a subscriber in the Amazon Simple Notification Service Developer Guide.

Polling the SQS queue for notifications

Finally, you need to define a service that continually polls the queue, looks for messages, and handles them accordingly.

Accessing the AWS Marketplace Metering and Entitlement Service APIs

This section outlines the process of integrating with the AWS Marketplace Metering Service or AWS Marketplace Entitlement Service, used to ensure your billing and reporting of customer usage of your SaaS products is accurate. It's assumed that you've submitted a SaaS subscriptions product or a SaaS contracts product that has been published to a limited state. In a limited state, you can use your test accounts to verify proper configuration and function but your product is not available publicly.

Topics

- Metering for usage (p. 130)
- Checking entitlements (p. 131)
For information about setting up the AWS CLI, along with credentials, see Configuring the AWS CLI in the AWS Command Line Interface User Guide. If you're new to the AWS Python SDK, see the Boto 3 Quickstart.

Metering for usage

For software as a service (SaaS) subscriptions, you meter for all usage, and then customers are billed by AWS based on the metering records that you provide. For SaaS contracts, you only meter for usage beyond a customer's contract entitlements. When your application meters usage for a customer, your application is providing AWS with a quantity of usage accrued. Your application meters for the pricing dimensions that you defined when you created your product, such as gigabytes transferred or hosts scanned in a given hour. For example, if you charge based on the amount of data sent into your application, you can measure the amount of data and send a corresponding metering record once an hour. AWS calculates a customer's bill using the metering data along with the prices that you provided when you created your product.

**Note**

Optionally, you can split the usage across properties that you track. These properties are exposed to the buyer as tags. These tags allow the buyer to view their costs split into usage by the tag values. For example, if you charge by the user, and users have a "Department" property, you could create a usage allocations with tags that have a key of "Department", and one allocation per value. This does not change the price, dimensions, or the total usage that you report, but allows your customer to view their costs by categories appropriate to your product.

We recommend that you send a metering record every hour to give customers as much granular visibility into their usage and costs as possible. If you aggregate usage in time periods greater than an hour (for example, one day), continue sending metering records every hour and record a quantity of 0 if there is no usage to report for that hour. Report usage to AWS on an hourly basis for all of your customers, in batches of up to 25 at a time.

AWS can only bill customers for usage of your product upon receiving metering records from you. You're responsible for ensuring that your product's metering records are successfully transmitted and received. You can use AWS CloudTrail to verify the record or records that you send are accurate. You can also use the information to perform audits over time. For more information, see Logging AWS Marketplace API calls with AWS CloudTrail (p. 215).

Configure your product to meter usage

You use the BatchMeterUsage operation in the AWS Marketplace Metering Service to deliver metering records to AWS. Keep the following in mind:

- We require sellers to use batching by using the BatchMeterUsage operation.
- We deduplicate metering requests on the hour.
  - Requests are deduplicated per product/customer/hour/dimension.
  - You can always retry any request, but if you meter for a different quantity, the original quantity is billed.
  - If you send multiple requests for the same customer/dimension/hour, the records are not aggregated.
- Your metering records contain a timestamp that can't be later than 1 hour in the past.
- BatchMeterUsage payloads must not exceed 1MB. Choose the number of usage records to send in a BatchMeterUsage request so that you don't exceed the size of the payload.
- The AWS Marketplace Metering Service is available in 14 AWS Regions. By default, the US East (N. Virginia) Region is enabled for SaaS metering products when you request your product. If you intend
Checking entitlements

If your product is a SaaS contracts product, your product calls the AWS Marketplace Entitlement Service to retrieve the customer’s entitlement using the GetEntitlements operation. Your product should verify subsequent usage on that account against the AWS Marketplace Entitlement Service. For example, if the customer provisions 10 users on the account, your product should check the AWS Marketplace Entitlement Service for entitlement to that capacity.

To verify a customer's entitlement to your product, use the GetEntitlements operation in the AWS Marketplace Entitlement Service. The AWS Marketplace Entitlement Service is available only in the US East (N. Virginia) Region, accessible through entitlement.marketplace.us-east-1.amazonaws.com.

GetEntitlements accepts a customer identifier and dimension as filters. ProductCode is a required parameter. The operation returns a paginated list of entitlements. The result has an ExpirationDate field that shows the minimum period of time that the entitlement is valid for. If the customer has set up automatic renewal, the date in the ExpirationDate field is the renewal date.

For code samples, see Code examples (p. 134).

Retrieving entitlement on user actions

The following examples can help you better understand the process for retrieving entitlement on user actions.

Example: User-based product

You offer a product that allows a number of accounts to exist for a given customer. The customer can visit a dashboard to provision new users (for example, to assign credentials). When the customer provisions a new user, your product calls GetEntitlements to verify that the capacity exists. If it does not, you can call the AWS Marketplace Metering Service to bill for additional users.
Example: Data storage product

You offer a product that enables customers to store a certain amount of data in encrypted or unencrypted form. The customer can view a dashboard that displays the amount of data existing and allocated in your product. Your dashboard retrieves the allocation amount through `GetEntitlements`.

SaaS product integration checklist

Before your SaaS product goes live, use this checklist to verify that you have completed the required configuration.

<table>
<thead>
<tr>
<th>Category</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>Submitted a seller registration form with the desired AWS account for AWS Marketplace usage.</td>
</tr>
<tr>
<td>Access</td>
<td>Completed the seller registration, including terms and conditions, bank account, and W8 or W9 tax form.</td>
</tr>
<tr>
<td>Access</td>
<td>Configured cross-account roles for the registered AWS Marketplace account.</td>
</tr>
<tr>
<td>Product</td>
<td>Completed the product request form in the AWS Marketplace Management Portal.</td>
</tr>
<tr>
<td>Product</td>
<td>Provided AWS account IDs for testing in the Notes tab of the Create product wizard in the AMMP.</td>
</tr>
<tr>
<td>Product</td>
<td>Provided a URL of the EULA in .txt format in the Products tab.</td>
</tr>
<tr>
<td>Product</td>
<td>Received your product code and Amazon SNS topic information from AWS Marketplace.</td>
</tr>
<tr>
<td>Product</td>
<td>Subscribed to the Amazon SNS topic and created an Amazon SQS queue to subscribe to the Amazon SNS topic.</td>
</tr>
<tr>
<td>Billing Solution</td>
<td>Validated you can send metering records to the BatchMeterUsage operation each hour for each customer for SaaS subscriptions products. Can send metering records for additional usage by each customer for SaaS contracts products.</td>
</tr>
<tr>
<td>Billing Solution</td>
<td>Validated you can verify customer entitlements from the AWS Marketplace Entitlement Service for SaaS contracts products.</td>
</tr>
<tr>
<td>Billing Solution</td>
<td>Validated that the costs appear as expected on bills generated for test accounts.</td>
</tr>
<tr>
<td>Billing Solution</td>
<td>Tested for situations such as invalid customer IDs and canceled subscriptions.</td>
</tr>
<tr>
<td>Product</td>
<td>Submitted the product request back to AWS Marketplace for publishing.</td>
</tr>
<tr>
<td>Category</td>
<td>Requirements</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Registration</td>
<td>Implemented an HTTPS registration page that can accept HTTP POST requests.</td>
</tr>
<tr>
<td>Registration</td>
<td>Validated you can accept new customer registrations.</td>
</tr>
<tr>
<td>Registration</td>
<td>Validated you are not storing the registration token in a cookie.</td>
</tr>
<tr>
<td>Registration</td>
<td>Validated you are using ResolveCustomer to obtain the ProductCode and CustomerIdentifier from the AWS token.</td>
</tr>
<tr>
<td>Registration</td>
<td>Validated you can resolve the registration token received from AWS with no delays.</td>
</tr>
<tr>
<td>Registration</td>
<td>Tested that you aren't blocked from registering with email services addresses such as Gmail.</td>
</tr>
<tr>
<td>Registration</td>
<td>Tested that you can accept incomplete registrations and multiple registration attempts.</td>
</tr>
<tr>
<td>Subscription</td>
<td>Test that you can handle unsubscribe-pending and unsubscribe-success messages.</td>
</tr>
<tr>
<td>Subscription</td>
<td>Validated that you send final metering records within an hour of receiving an unsubscribe-pending message.</td>
</tr>
<tr>
<td>Security</td>
<td>Validated the AWS root account doesn't have API keys, has a strong password, and is associated with a hardware multi-factor authentication (MFA) device. All administrative access is through identities created with AWS Identity and Access Management (IAM). No shared accounts.</td>
</tr>
<tr>
<td>Security</td>
<td>Validated that IAM roles are used for all programmatic Amazon Elastic Compute Cloud (Amazon EC2) access. Credentials aren't hard-coded into scripts, headers, or source code.</td>
</tr>
<tr>
<td>Security</td>
<td>Validated you maintain comprehensive logging and log consolidation.</td>
</tr>
<tr>
<td>Security</td>
<td>Verified you have well-defined public and private subnet boundaries that isolate application services and access to database and file systems. Distinct data class definitions that demarcate sensitive data and segregate public and private data.</td>
</tr>
<tr>
<td>Security</td>
<td>Verified you have private data encryption in transit and at rest with scheduled key rotation.</td>
</tr>
<tr>
<td>Security</td>
<td>Validated you have security incident tools and access in place and routinely scheduled incident response exercises that accommodate timely investigation and recovery.</td>
</tr>
</tbody>
</table>
Reporting

AWS Marketplace produces reports for your SaaS products that include data on subscribers, financials, usage, and taxes. For more information, see the section called “Seller reports” (p. 153). The following table summarizes how financials for SaaS products are reported.

<table>
<thead>
<tr>
<th>Report</th>
<th>SaaS content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daily business report</td>
<td>Upfront contract charges appear in the Fees section.</td>
</tr>
<tr>
<td></td>
<td>Metered usage charges appear in the Usage section.</td>
</tr>
<tr>
<td>Monthly revenue report</td>
<td>Upfront contract charges appear in the Annual subscriptions section.</td>
</tr>
<tr>
<td></td>
<td>Metered usage charges appear in the Billing and revenue data section.</td>
</tr>
<tr>
<td>Sales compensation report</td>
<td>Upfront contract charges and monthly additional usage charges appear as separate line items.</td>
</tr>
<tr>
<td>Customer subscriber report</td>
<td>New SaaS contracts appear in the Annual subscriptions section.</td>
</tr>
<tr>
<td></td>
<td>New SaaS subscriptions appear in the Hourly/monthly subscriptions section.</td>
</tr>
</tbody>
</table>

Code examples

The following code examples are provided to help you integrate your SaaS product with the AWS Marketplace APIs required for publishing and maintaining your product.

Topics

- ResolveCustomer code example (p. 135)
- GetEntitlement code example (p. 135)
- BatchMeterUsage code example (p. 136)
ResolveCustomer code example

The following code example is relevant for all pricing models. The Python example exchanges a x-amzn-marketplace-token token for a customerID. This code would run in an application on your registration website, when they are redirected there from the AWS Marketplace Management Portal. The redirect is a POST request that includes the token.

For more information on ResolveCustomer, see ResolveCustomer in the AWS Marketplace Metering Service API Reference.

```python
# Import AWS Python SDK and urllib.parse
import boto3
import urllib.parse as urlparse

# Resolving Customer Registration Token
formFields = urlparse.parse_qs(postBody)
regToken = formFields['x-amzn-marketplace-token']

# If regToken present in POST request, exchange for customerID
if (regToken):
    marketplaceClient = boto3.client('meteringmarketplace')
    customerData = marketplaceClient.resolve_customer(regToken)
    productCode = customerData['ProductCode']
    customerID = customerData['CustomerIdentifier']

    # TODO: Store customer information
    # TODO: Validate no other accounts share the same customerID
```

Example response

```
{
    'CustomerIdentifier': 'string',
    'ProductCode': 'string'
}
```

GetEntitlement code example

The following code example is relevant for SaaS products with the contract and SaaS contract with consumption pricing model. The Python example verifies that a customer has an active entitlement.

For more information on GetEntitlement, see GetEntitlement in the AWS Marketplace Entitlement Service API Reference.

```python
# Import AWS Python SDK
import boto3

marketplaceClient = boto3.client('marketplace-entitlement')

# Filter entitlements for a specific customerID
# productCode is supplied after the AWS Marketplace Ops team has published
# the product to limited
# customerID is obtained from the ResolveCustomer response
entitlement = marketplaceClient.get_entitlements({
    'ProductCode': 'productCode',
    'Filter': {
```
'CUSTOMER_IDENTIFIER': [
    'customerID',
],
'NextToken': 'string',
'MaxResults': 123
))

# TODO: Verify the dimension a customer is subscribed to and the quantity,
# if applicable

**Example response**

The returned value will correspond with the dimensions created when you created the product in the AWS Marketplace Management Portal.

```json
{
  "Entitlements": [
    {
      "CustomerIdentifier": "string",
      "Dimension": "string",
      "ExpirationDate": number,
      "ProductCode": "string",
      "Value": {
        "BooleanValue": boolean,
        "DoubleValue": number,
        "IntegerValue": number,
        "StringValue": "string"
      }
    }
  ],
  "NextToken": "string"
}
```

**BatchMeterUsage code example**

The following code example is relevant for SaaS subscription and contract with consumption pricing models, but not for SaaS contract products without consumption. The Python example sends a metering record to AWS Marketplace to charge your customers for pay-as-you-go fees.

```python
# NOTE: Your application will need to aggregate usage for the customer for the hour and set the quantity as seen below. AWS Marketplace can only accept records for up to an hour in the past.
# productCode is supplied after the AWS Marketplace Ops team has published the product to limited
# customerID is obtained from the ResolveCustomer response

# Import AWS Python SDK
import boto3

usageRecord = [
    {
        'Timestamp': datetime(2015, 1, 1),
        'CustomerIdentifier': 'customerID',
        'Dimension': 'string',
        'Quantity': 123
    }
]
```
Using AWS PrivateLink with AWS Marketplace

AWS Marketplace supports AWS PrivateLink, a technology that allows you to use the Amazon network to provide buyers with access to products you sell through AWS Marketplace. This document outlines the process for configuring and delivering your products through an Amazon Virtual Private Cloud (VPC) endpoint using AWS PrivateLink technology.

In this document, we assume that you have working knowledge of several AWS services and the AWS Marketplace environment.

Introduction

As an AWS Marketplace seller, you can provide buyers access to your service through an Amazon VPC endpoint. This approach provides buyers with access to your service across the Amazon network using AWS PrivateLink technology. If you use AWS Marketplace to create and deliver this offering, buyers can discover your service in AWS Marketplace. Your buyers can also find your product in the list of available services for creating a VPC endpoint.
A VPC endpoint is a virtual device that enables AWS customers to create a private connection between their VPC and another AWS service without requiring access over the internet, through a NAT device, a VPN connection, or AWS Direct Connect. You can create an endpoint service through AWS Marketplace that makes it possible for buyers to use this technology to connect to your service. This connection method is more secure for your buyers because they access your service through the Amazon private network rather than through the Internet.

For each region where you want to offer your service, you create or use existing resources to configure a VPC, set up your service instances, set up a network load balancer, and register your services with the network load balancer by creating a service endpoint. After you complete those steps and test your offering, you provide your configuration information to the the AWS Marketplace Seller Operations team.

AWS recommends that provide a private DNS name that your buyers can use when they create VPC endpoints.

When buyers create their VPC endpoints, they have the option to enable a private DNS name. By choosing this option, the buyer’s VPC service configures a private hosted zone. If you provide the private DNS name, buyers can use it when configuring VPC endpoints to connect to your service. In the buyer’s private hosted zone, the private DNS name (api.example.com) will point to the randomly generated DNS name(s) (vpce-11111111111111111-yyyyyyyy.api.vpce.example.com) created for your endpoint service(s). The buyer’s EC2 instances call the same unified DNS name (api.example.com) across different VPCs. Also, if public and private DNS names are same, the buyer can use the same public name when accessing your service from within or outside of the VPC.

For assistance with making your service available through AWS Marketplace, you can contact the AWS Marketplace Seller Operations team. When an AWS Marketplace buyer subscribes to your service and creates a VPC endpoint, your service is shown under Your AWS Marketplace Services. The MCO team uses the user-friendly DNS name for ease of discovery of your service when creating the VPC endpoint.

Your product is created as a software as a service (SaaS) product. Metering and billing is the same as with other AWS Marketplace SaaS products.

**Configuring your product**

To configure your product to be available through an Amazon VPC endpoint:

1. Create or use an existing Amazon VPC.
2. Create (or use existing) Amazon EC2 instance(s) for your product.
3. Create a network load balancer in each of the regions where you offer your product. AWS recommends that you include all Availability Zones (AZs) for a region.
4. Use the Amazon VPC console, the CLI, or supported SDKs to create a VPC endpoint service.
5. Verify that you can access the service through the network load balancer.
6. Request a certificate from AWS Certificate Manager (ACM) for your user-friendly DNS name. Before ACM issues a certificate, it validates that you own or control the domain names in your certificate request.

7. Delegate the subdomain of your user-friendly DNS name, such as api.vpce.example.com, to the name servers provided to you by the MCO team. In your DNS system, you must create a name server (NS) resource record to point this subdomain to the Amazon Route 53 name servers provided by the MCO team so that DNS names (such as vpce-0ac6c347a78c90f8.api.vpce.example.com) are publicly resolvable.

8. Allow access to your buyers' AWS accounts.

   **Note:** You can use a supported SDK or this CLI command to automate access to accounts: `aws vpcev2 modify-vpc-endpoint-service-permissions --service-id vpce-svc-0123456789abcdef1 --add-allowed-principals arn:aws:iam::111111111111:root arn:aws:iam::222222222222:root.

---

**Submitting your product to AWS Marketplace**

During the process of publishing your service to AWS Marketplace, you work with the AWS Marketplace Seller Operations team. To submit your PrivateLink-enabled product:

1. Email the following information to the AWS Marketplace Seller Operations team:
   a. The endpoint and the AWS account used to create the endpoint. The endpoint is similar to this: com.amazonaws.vpce.us-east-1.vpce-svc-0daa010345a21646
   b. The user-friendly DNS name for your service. This is the DNS name that AWS Marketplace buyers use to access your product.
   c. The AWS account that you used to request certificates and the private DNS name buyers use to access the VPC endpoint.

   The AWS Marketplace MCO team verifies your company’s identity and the DNS name to use for the service you are registering (such as api.vpce.example.com). After verification, the DNS name overrides the default base endpoint DNS name.

**Buyer access to VPC endpoints**

AWS Marketplace buyers who are creating a VPC endpoint can discover your service in these situations:

- You followed the seller processes described earlier on this page to create or use an existing product.
- The buyer subscribes to your service.
- You added the buyer's AWS account to your list of allowed accounts.

When the buyer creates the VPC endpoint, they have the option to associate a private hosted zone with their VPC. The hosted zone contains a record set for the default private DNS name for the service that resolves to the private IP address of the endpoint network interfaces in their VPC.

Any buyer-hosted endpoint, including AWS Marketplace services, can provide permissions to all accounts (the "*" permission). However, when you use this approach, the services aren’t included in the Describe calls or console unless you search by the service name. To display the services in the Describe calls, the buyer’s AWS account must be explicitly added to the allow list by the service.

To access your service, buyers do the following:

1. Discover and subscribe to your service on AWS Marketplace.
2. Use the AWS Command Line Interface (AWS CLI), API, or the Amazon VPC console to discover your service and then establish a VPC endpoint to connect to your service in the subnets and AZs they use.
The endpoints are shown as elastic network interfaces in the subnets. Local IP addresses and region and zonal DNS names are assigned to the endpoints.

<table>
<thead>
<tr>
<th>Client-side DNS name</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regional</td>
<td>Vpce&lt;0dc9a211a78c90f8&gt;.api.vpce.example.com</td>
</tr>
<tr>
<td>IAD2 (1a)</td>
<td>us-east-1a-Vpce&lt;0dc9a211a78c90f8&gt;.api.vpce.example.com</td>
</tr>
<tr>
<td>IAD2 (1b)</td>
<td>us-east-1b-Vpce&lt;0dc9a211a78c90f8&gt;.api.vpce.example.com</td>
</tr>
</tbody>
</table>

If you provided a default private DNS name and the buyer chooses Enable Private DNS Name (associated a private hosted zone) when creating a VPC endpoint, the buyer sees the regional default private DNS name to connect to your service.

<table>
<thead>
<tr>
<th>Name</th>
<th>Alias</th>
<th>Alias hosted zone ID</th>
<th>(Notes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>api.example.com</td>
<td>vpce&lt;0dc9a211a78c90f8&gt;&gt;Z00AABBCCDD</td>
<td>api.vpce.example.com</td>
<td>IAD1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IAD2</td>
</tr>
</tbody>
</table>

**Appendix: Checklists**

Use the following checklists to ensure that you configure and test your product before you submit it to the MCO team.

**Product creation checklist**

- Create (or use an existing) VPC and then configure it.
- Create and configure a network load balancer within the VPC.
- Register your service with your network load balancer by creating a VPC endpoint service.
- Provide the AWS account ID you used to configure the VPC endpoint to MCO.
- Provide the default endpoint service name (for example, com.amazonaws.vpce.us-east-1.vpce-svc-0bbb070044a2164) to MCO.
- Provide a user-friendly service DNS name (required) to override the randomly generated service DNS name. Request SSL certificates from ACM for the subdomain used for your user-friendly service DNS name. Provide these certificates and the AWS account ID you used to request them to the MCO team.
- Recommended: Provide a private DNS name.
- Create a process to inform and allow your AWS Marketplace buyers the option to connect to your service using AWS PrivateLink technology. Add AWS account IDs for your buyers to your allowed list of accounts.

**Product testing**

- Verify that your service is configured and discoverable.
- Verify that your service is discoverable over the network load balancer.
- Verify that a buyer can create a VPC endpoint and access your service. Use an AWS account you own that is not the account you used to set up your service.
Data products

AWS Data Exchange is a service that makes it easy for AWS customers to securely exchange file-based data sets in the AWS Cloud. As a provider, AWS Data Exchange eliminates the need to build and maintain any data delivery, entitlement, or billing technology. Providers in AWS Data Exchange have a secure, transparent, and reliable channel to reach AWS customers and grant existing customers their subscriptions more efficiently. The process for becoming an AWS Data Exchange provider requires a few steps to determine eligibility.

A data product has the following parts:

- **Product details** – This information helps potential subscribers understand what the product is. This includes a name, descriptions (both short and long), a logo image, and support contact information. Product details are filled out by providers.

- **Product offer(s)** – In order to make a product available on AWS Data Exchange, providers must define a public offer. This includes the prices and durations, data subscription agreement, refund policy, and the option to create custom offers.

- **Data set(s)** – A product can contain one or more data sets. A data set is a dynamic set of file-based data content. Data sets are dynamic and are versioned using revisions. Each revision can contain multiple assets.

For more information, including eligibility requirements, see Providing Data Products on AWS Data Exchange in the AWS Data Exchange User Guide.
Submitting your product for publication

You use the product submission process to make your products available on AWS Marketplace. Products can be quite simple, for example a single Amazon Machine Image (AMI) that has one price structure. Or, products can be quite complicated, with multiple AMIs, AWS CloudFormation templates, and complex pricing options and payment schedules. You define your product offering and submit it through the AWS Marketplace Management Portal in one of two ways:

- Using the **Products** tab – For products that are less complex, you use the **Products** tab to completely define and submit your request.
- Using the **Assets** tab – For products that are more complex and require more definition, you download a product load form (PLF), add product details, and then upload the completed form using the **File upload** option.

**Note**

Data product providers must use the AWS Data Exchange console to publish products. For more information, see Publishing Products in the *AWS Data Exchange User Guide*.

We recommend that you start by using the **Products** tab to determine which approach to use. The following table lists configurations and the approach you use to submit your request. The first column is the pricing model for your product, and the other three columns are how the product is deployed to the customer.

<table>
<thead>
<tr>
<th>Pricing model</th>
<th>Products launched using single-node AMI</th>
<th>Products launched with AWS CloudFormation</th>
<th>Products launched as software as a service (SaaS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bring Your Own License (BYOL)</td>
<td><strong>Products</strong> tab</td>
<td><strong>Assets</strong> tab</td>
<td></td>
</tr>
<tr>
<td>Free</td>
<td><strong>Products</strong> tab</td>
<td><strong>Assets</strong> tab</td>
<td></td>
</tr>
<tr>
<td>Hourly</td>
<td><strong>Products</strong> tab</td>
<td><strong>Assets</strong> tab</td>
<td></td>
</tr>
<tr>
<td>Hourly with Annual</td>
<td><strong>Products</strong> tab</td>
<td><strong>Assets</strong> tab</td>
<td></td>
</tr>
<tr>
<td>Monthly</td>
<td><strong>Assets</strong> tab</td>
<td><strong>Assets</strong> tab</td>
<td></td>
</tr>
<tr>
<td>Hourly with Monthly</td>
<td><strong>Assets</strong> tab</td>
<td><strong>Assets</strong> tab</td>
<td></td>
</tr>
<tr>
<td>Usage (AWS Marketplace Metering Service)</td>
<td><strong>Assets</strong> tab</td>
<td><strong>Assets</strong> tab</td>
<td></td>
</tr>
<tr>
<td>SaaS Subscription</td>
<td><strong>Products</strong> tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SaaS Contract</td>
<td><strong>Products</strong> tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SaaS Legacy</td>
<td><strong>Assets</strong> tab</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You can submit products individually or, if you use a product load form, you can submit multiple products or product updates at the same time. You cannot submit multiple products at the same
Using the Products tab

To access the Products tab, log in to the AWS Marketplace Management Portal. From the Products tab, choose either Server, SaaS, or Machine learning, depending on the type of product you are managing. A dashboard for that product type appears that contains all of your current products. If you choose the Requests tab, the dashboard displays any outstanding requests you have and your completed request history. Once you start creating a new product request, you can save your work in progress, and if necessary, create your request in several different sessions.

When you are ready to submit your product request, the request is reviewed by the AWS Marketplace team. You can monitor the status of your request on the product page for the type of product you are requesting. For new products, after your request is approved for publication, you receive a limited listing URL that you can use to preview and approve your submission. Your product offer is not published until you approve the submission. When you request an update to an existing product, the update is published without the need for you to review and approve the change. This includes adding or removing versions, and metadata changes.

You track the status of your requests under the Requests tab. The status will be one of the following:

- **Draft** – You have started the request process but have not submitted your request.
- **Submitted** – You have completed and submitted your request, and it is under review.
- **Action Required** – The AWS Marketplace team has reviewed your request and needs more information.
- **Approval Required** – The AWS Marketplace team has created the limited listing URL for your product. You must review and either approve or reject the URL before AWS Marketplace will publish. If you approve, the status changes to Publishing Pending while the site gets published. If you reject, the status returns to Draft so you can modify the request.
- **Publishing Pending** – You have approved the mock-up of your request and AWS Marketplace is publishing your product.
- **Expired** – You started the request process but did not complete it within six months, so the request expired.

If you have an entry with a status of Submitted, you can retract the submission. If you have an entry with a status of Draft, you can delete the request. This will allow you to start over. When you delete a Draft entry, the entry is moved to the Request History tab.

To add your product in the AWS GovCloud (US) AWS Region, you must have an active AWS GovCloud (U.S.) account and comply with the AWS GovCloud (US) requirements, including export control requirements.

Company and product logo requirements

Your company logo and the logo for your products must conform to the following AWS Marketplace guidelines so that the user experience is uniform when browsing AWS Marketplace:

- **Product logo specifications** – Your product logo image should have a transparent or white background and be 120 to 640 pixels in size, with a 1:1 or 2:1 (wide) ratio.

- **Company logo specifications** – Your company logo image should have a transparent background and be 220 x 220 pixels in size, allowing for 10 pixels of padding on each side within.
Requirements for submitting paid repackaged software

If you are submitting a paid listing of either a repackaged open-source software (for example, open source AMI or container products with paid support), or software that was originally created by a vendor other than you (for example, reselling an AMI with Windows operating system), the following requirements must be met before submission:

- The product title must indicate the value added by your repackaging. Examples of product titles include: Secure Hardened <Product>, <Product> with added packages, <Product1> on <Product2>.
- The product title must not contain any other language that is not otherwise supported with documentation. For example, the product title may not use the words certified, original, or free unless these are substantiated in the product details that you provide.
- The product short description must include a clear statement summarizing the product charges. The short description must begin with the phrase This product has charges associated with it for... For example, if a product includes charges for support from the seller, then the product description should state: This product has charges associated with it for seller support.
- The product logo must be same as the company logo which was used during your seller registration process. If you are using a logo other than your own, explicit permission from the original software's vendor is required for using the original product logo. If explicit permission is obtained, a link to that documentation must be included in the notes section of the change request (or in the Enter a brief description field of the File Uploads page when using the product load form).
- For AMI products, the AMI name must not be reused from the original product. The AMI name must begin with the seller name and follow this format: [Seller Name] [name-given-to-ami].

If the paid listing is for a standalone software product that was not created by your company and there is no intellectual property added to the product (for example, bundling additional software libraries or adding special configuration) then, along with the earlier requirements, the following requirements must also be met:

- Product title must include the seller name (along with the value added, as described earlier). The seller name is the name used during seller registration. For example, <Product> with maintenance support by <seller>.
- The first line of the product's long description must begin with the phrase This is a repackaged software product wherein additional charges apply for... (or, if it's open source, This is a repackaged open source software product wherein additional charges apply for...). Then, the long description must include a clear statement summarizing what you are charging for, as well as additional details describing those features, for example, the response times and severity descriptions of your maintenance. For example, the long description of an open source product charging for additional support might start as: This is a repackaged open source software product wherein additional charges apply for support with {SLA Details}.

AWS CloudFormation-launched product (free or paid) or usage-based paid AMI product

Use a product load form (PLF) to submit products that AWS Marketplace customers launch by using AWS CloudFormation templates. The PLF is available through the AWS Marketplace Management Portal (AMMP).
Submitting your product

1. From the AMMP, download the Product Load Form (PLF) for your product.
2. Add your product definition, which includes product information (title, description, highlights), technical information (AMI_ID, Regions, instance types, OS), and pricing details (pricing model, Free Trial).
3. Submit your PLF following the instructions under the Instructions table of the spreadsheet.

The AWS Marketplace team reviews your product for policy and security compliance, software vulnerabilities, and product usability. If there are any questions or issues with a request, the AWS Marketplace team will contact you via an email message to discuss your request. Once approved, a mock-up of your product's page is created. After you review the page, you accept or reject the mock-up. Once approved, we add the page to the AWS Marketplace.

Updating your product

For products that you created by using the Product Load Form (PLF), you also use the PLF to make changes to those products. You can make changes to the original PLF you completed or, if it's not available, you can start with a new PLF. Just like using the Products tab, you can add a new version, remove existing versions, and update pricing, instance types, region availability, and metadata. To make an update, you prepare any updated product the same way you prepare a new product. After the product update is prepared, follow these steps:

1. Use your existing PLF or, from the AWS Marketplace Management Portal, under the Assets tab, choose File upload. Under Product load forms and seller guides, you can download the PLF for your product.
2. Update your product submission in the PLF.
4. On the File Uploads page, upload your updated PLF and any AWS CloudFormation templates. The file uploader provides a secure transfer mechanism and a history of submitted files. The uploader automatically notifies the AWS Marketplace team to begin processing your request. Include a description of the submission (adding new version, changing price, changing metadata, and so forth).

Your product submission is reviewed for policy and security compliance, software vulnerabilities, and product usability. If there are any questions or issues with a request, the AWS Marketplace team will contact you via an email message. Updates to existing product pages are processed and released directly without additional reviews.

Product changes and updates

Sellers can submit changes to their product at any time, and they will be processed as described earlier. However, some changes can only be made every 90 or 120 days, or when pending changes are in place. Examples include price changes and AWS Region or instance type changes. Common changes include:

- **New Version** – New versions of the software and rollouts of patches or updates. At your request, we can notify customers who have subscribed to your AWS Marketplace content about the availability of new versions or send upgrade instructions on your behalf.
- **Metadata change** – Changes to product information (Description, URLs, and Usage Instructions).
- **Pricing Change** – A change to the pricing amount. A notification to current customers is sent after the request is complete.
• **Pricing Model Change** – A change to the pricing model (for example, Hourly, Free, Hourly_Annual). Not all pricing model changes are supported, and all requests to change models must be reviewed and approved by the AWS Marketplace team. Any change from a free to a paid model presents significant impact to existing customers. An alternative is to propose a new product with additional features and encourage current customers to migrate.

• **Region or Instance change** – Adding or removing instances types or Regions.

• **Product takedown** - Remove a product page from AWS Marketplace to prevent new customers from subscribing. A notification to current customers is sent after the request is complete.

### Timing and expectations

While we strive to process requests as quickly as possible, requests can require multiple iterations and review by the seller and AWS Marketplace team. Use the following as guidance for how long it will take to complete the process:

- Total request time normally takes 2–4 weeks of calendar time. More complex requests or products can take longer, due to multiple iterations and adjustments to product metadata and software.

- Review and processing of requests typically requires 3 business days. We will notify you if there are any issues that require additional action.

- We require a completed product request and AMI at least 45 days in advance of any planned events or releases, so we can prioritize the request accordingly.

If you have any questions about your request, contact the **AWS Marketplace Seller Operations** team.

### Submitting AMIs to AWS Marketplace

All AMIs built and submitted to AWS Marketplace must adhere to all product policies. We suggest a few final checks of your AMI prior to submission:

- Remove all user credentials from the system; for example, all default passwords, authorization keys, key pairs, security keys or other credentials.

- Ensure that root login is disabled or locked. Only sudo access accounts are allowed.

- If you are submitting an AMI to be deployed into the AWS GovCloud (US) Region, you need to have an active AWS GovCloud account and agree to the [AWS GovCloud Requirements](https://aws.amazon.com/govcloud-us/policy/), including applicable export control requirements.

### AMI self-service scanning

Self-service AMI scanning is available within the AWS Marketplace Management Portal. With this feature, you can initiate scans of your AMIs and receive scanning results quickly—typically in less than an hour—with clear feedback in a single location.

**To begin sharing and scanning your AMI with self-service scanning**

2. Select the AMI to share.
3. View your scan results.
After your AMI has successfully been scanned, you can follow the current process to submit it to the AWS Marketplace Seller Operations team by uploading your product load form (PLF). If you have any issues, contact the AWS Marketplace Seller Operations team.

To include your AMI in the self-service scanning list, the AMI must be in the `us-east-1` (N. Virginia) Region and owned by your AWS Marketplace seller account. If you need to grant other accounts access to the AWS Marketplace Management Portal, you must register those accounts as sellers. For more information, see Seller registration process (p. 5).

**AMI cloning and product code assignment**

After your AMI is submitted, AWS Marketplace creates cloned AMIs for each Region that you have indicated that software should be available in. During this cloning and publishing process, AWS Marketplace attaches a product code to the cloned AMIs. The product code is used to both control access and to meter usage. All submissions must go through this AMI cloning process.

**Final checklist**

To help avoid delays in publishing your product, use this checklist before you submit your product request.

**Product usage**

- Production-ready.
- Does not restrict product usage by time or other restrictions.
- Compatible with 1-click fulfillment experience.
- Everything required to use the product is contained within the software, including client applications.
- Default user uses a randomized password and/or creation of initial user requires verification that the buyer is authorized to use the instance using a value unique to the instance such as instance ID.

**For free or paid products**

- No additional license is required to use the product.
- Paid repackaged software meets the AWS Marketplace Requirements for submitting paid repackaged software (p. 144).
- Buyer does not have to provide personally identifiable information (for example, an email address) to use the product.

**AMI preparation**

- Use hardware virtual machine (HVM) virtualization and 64-bit architecture.
- Does not contain any known vulnerabilities, malware, or viruses.
- Buyers have operating system-level administration access to the AMI.
- Run your AMI through AMI Self-Service Scanning.

**For Windows AMIs**

- Use the most recent version of `Ec2ConfigService`, as described in Configuring a Windows Instance Using EC2Config Service.
- The `Ec2SetPassword`, `Ec2WindowsActivate`, and `Ec2HandleUserData` plugins are enabled, as described in Configuring a Windows Instance Using EC2Config Service.
• No Guest Accounts or Remote Desktop Users are present.

For Linux AMIs

• Root login is locked and disabled.
• No authorized keys, default passwords, or other credentials are included.
• All required fields are completed.
• All values are within specified character limits.
• All URLs load without error.
• Product image is at least 110px wide and between a 1:1 and 2:1 ratio.
• Pricing is specified for all enabled instance types (for hourly, hourly_monthly, and hourly_annual pricing models).
• Monthly pricing is specified (for hourly_monthly and monthly pricing models).

If you have any questions or comments about automated AMI building, contact the AWS Marketplace Seller Operations team.
Marketing your product

You can contribute to the success of your product by driving awareness of AWS Marketplace and by driving traffic directly to your product pages on AWS Marketplace. The following provides information and support to help you market the product or products that you have listed on AWS Marketplace. For more information, see AWS Marketplace Go-to-Market Program Guide.

Announcing your product's availability

We encourage you to broadly announce the availability of your product on AWS Marketplace. You can do this via press releases, tweets, blogs, or any other preferred media channels. We have provided sample text that you can include, along with guidelines and instructions for using our trademarks and issuing press releases.

We will review your blogs, tweets, and other non-press release announcements before going public to ensure consistency with AWS messaging and brand guidelines or voice. Submit your request for review to your AWS account manager. The review takes up to 10 business days to complete. Notify us when you post any tweets, blogs, or press releases, and we will do our best to repost to increase their visibility.

AWS Marketplace messaging

In your customer communications you might want to describe the purpose, goals, and benefits of purchasing your product using AWS Marketplace. Use the following messaging when referring to AWS Marketplace.

What is AWS Marketplace?

AWS Marketplace is an online store that makes it easy for customers to find, compare, and immediately start using the software and services that run on AWS. Visitors to AWS Marketplace can use 1-Click deployment to quickly launch preconfigured software and pay only for what they use, by the hour or month. AWS handles billing and payments, and software charges appear on the customer’s AWS bill.

Why would a customer shop on AWS Marketplace?

Finding and deploying software can be challenging. AWS Marketplace features a wide selection of commercial and free IT and business software, including software infrastructure such as databases and application servers, IoT solutions, developer tools, and business applications, from popular sellers. AWS Marketplace enables customers to compare options, read reviews, and quickly find the software they want. Then they can deploy it to their own Amazon Elastic Compute Cloud instance using 1-Click or using the AWS Marketplace Management Portal.

Software prices are clearly posted on the website and customers can purchase most software immediately, with payment instruments already on file with Amazon Web Services. Software charges appear on the same monthly bill as AWS infrastructure charges.

Why would software or SaaS sellers sell on AWS Marketplace?

With AWS Marketplace, software and software as a service (SaaS) sellers with offerings that run on AWS can benefit from increased customer awareness, simplified deployment, and automated billing.
AWS Marketplace helps software and SaaS sellers of software and services that run on AWS find new customers by exposing their products to some of the hundreds of thousands of AWS customers, ranging from individual software developers to large enterprises.

Selling on AWS Marketplace enables independent software vendors (ISVs) to add hourly billing for their software without undertaking costly code changes. They simply upload an Amazon Machine Image (AMI) to AWS and provide the hourly cost. Billing is managed by AWS Marketplace, relieving sellers of the responsibility of metering usage, managing customer accounts, and processing payments, leaving software developers more time to focus on building great software.

Additionally, customers benefit from the ability to easily deploy preconfigured images of the software, simplifying onboarding for new customers.

Reviews on AWS Marketplace

AWS Marketplace provides the ability for customers to submit reviews on your product. We also provide the ability for syndicated reviewers such as G2, a business-to-business marketplace that curates independent product reviews, to integrate their syndicated reviews on AWS Marketplace.

AWS Marketplace customer reviews must meet the review guidelines listed in the user guide for buyers. Review submissions are not released on AWS Marketplace until after the submission is reviewed to verify it meets our review criteria. For more information on review guidelines, see Product Reviews. Syndicated review organizations use their own unique processes to validate their reviews and aren't reviewed by AWS Marketplace before release. If you think that a syndicated review on your product doesn't meet the product review guidelines, or if you think a review on your product contains objectionable content, contact the seller operations team.

The reviewer can also provide a star rating for your product based on a five-star rating system. The ratings are averaged to give the overall star rating for your product. Syndicated reviews can also include a star rating, but star ratings from syndicated reviews are not averaged in with the AWS customer star ratings.

The following are additional key points about the product review feature:

- You can't have a product review removed from AWS Marketplace. However, you can leave a comment on any review as long as the comment meets the review criteria governing product reviews.
- If you think that a review doesn't meet the review guidelines or contains objectionable content, you can contact the seller operations team and describe your concern.
- AWS customers searching for products in AWS Marketplace can search and filter results based on ratings, verified reviews, and externally sourced reviews. AWS customers see the externally sourced ratings alongside AWS customer ratings in search results.
- Syndicated reviews for your product are automatically added to AWS Marketplace at no cost to you. Because reviews are automatically added, you don't need to submit a request to have a syndicated review added.
- If you don't have any syndicated reviews for your product, you can contact the syndicated reviewer and follow their process for getting your product reviewed. For example, with G2, you can visit their website and claim your product page to start their review process.

Linking to AWS Marketplace

Your company likely has a web presence where it describes and promotes your product. We encourage you to highlight that the product is available to run on AWS and can be purchased using AWS Marketplace. To simplify the process for your customers to discover and deploy your software, we have provided instructions for linking your customers to your product.
Using the AWS Marketplace logo

The AWS Marketplace logo is a way to easily tell your customers that your software runs on AWS and is available in AWS Marketplace. If you would like to promote your software in AWS Marketplace, download the archived folder (.zip file), which contains multiple color treatments and file formats.

Linking directly to your product on AWS Marketplace

You can send your customers directly to the product's information page on AWS Marketplace by including deep links on your website or collateral. Use the following example link structure for browser-based linking.

https://aws.amazon.com/marketplace/pp/ASIN

Replace ASIN segment of the URL with your product's ASIN.

Example

https://aws.amazon.com/marketplace/pp/B00635Y2IW

The ASIN appears in the URL when you search for your application on aws.amazon.com/marketplace. Alternatively, you can consult with your account manager to find the ASIN.

Note

Test the links before using them to make sure that they direct your customers to the correct page.

Press releases

We encourage you to announce your product's availability on AWS Marketplace through any channel you prefer. However, all press releases that reference AWS Marketplace must be reviewed and signed off on by Amazon before any publication or announcement is made. While we encourage you to make announcements, we can't support joint press releases with AWS Marketplace sellers. We will, on a case-by-case basis, support press releases with a quote from AWS. The quote must meet several conditions, including but not limited to: it announces a new product or service listed on AWS Marketplace or it includes a customer reference that uses AWS Marketplace.

All press releases must be drafted by you. We suggest the following headline: [Insert product name] Now Available on AWS Marketplace. Use the messaging in this document for consistency.

The press release should:

- Clearly and accurately describe how the announcement relates to Amazon.com
- Clarify your role on AWS and with customers
- Be customer-focused and emphasize the customer benefit

The press release should not:

- Use the terms partners, partnership, or alliance to describe the relationship. We prefer agreement, teamed, or relationship.
• Include a quote from an Amazon Web Services executive unless previously agreed upon.
• Include any sales projections or use .com by the merchant unless referring to the website in your company boilerplate.
• Refer to your organization as an associate of Amazon.com because this could be confused with Amazon Associates, our online affiliate program.
• Disclose proprietary information about Amazon.com or refer to our stock ticker symbol.

Have your press release reviewed by submitting it in text format to your account manager. Additionally, review the Amazon Web Services trademark guidelines before using any AWS trademarks. Guidelines specific to the AWS Marketplace trademark are in the following section.

AWS Marketplace trademark usage guidelines

These Guidelines apply to your use of the AWS Marketplace logo and trademark, (each the “Trademark” and collectively the “Trademarks”) in materials that have been approved in advance by Amazon.com, Inc. and/or its affiliates (“Amazon”). Strict compliance with these Guidelines is required at all times, and any use of a Trademark in violation of these Guidelines will automatically terminate any license related to your use of the Trademarks.

1. You may use the Trademark solely for the purpose expressly authorized by Amazon and your use must: (i) comply with the most up-to-date version of all agreement(s) with Amazon regarding your use of any of the Trademarks (collectively “Agreements”); (ii) comply with the most up-to-date version of these Guidelines; and (iii) comply with any other terms, conditions, or policies that Amazon may issue from time to time that apply to the use of the Trademark.
2. We will supply an approved Trademark image for you to use. You may not alter the Trademark in any manner, including but not limited to, changing the proportion, color, or font of the Trademark, or adding or removing any element(s) from the Trademark.
3. You may not use the Trademark in any manner that implies sponsorship or endorsement by Amazon other than by using the Trademark as specifically authorized under the Agreements.
4. You may not use the Trademark to disparage Amazon, its products or services, or in a manner which, in Amazon’s sole discretion, may diminish or otherwise damage or tarnish Amazon's goodwill in the Trademark.
5. The Trademark must appear by itself, with reasonable spacing between each side of the Trademark and other visual, graphic or textual elements. Under no circumstance should the Trademark be placed on any background which interferes with the readability or display of the Trademark.
6. You must include the following statement in any materials that display the Trademark: “AWS Marketplace and the AWS Marketplace logo are trademarks of Amazon.com, Inc. or its affiliates.
7. You acknowledge that all rights to the Trademark are the exclusive property of Amazon, and all goodwill generated through your use of the Trademark will inure to the benefit of Amazon. You will not take any action that is in conflict with Amazon’s rights in, or ownership of, the Trademark.

Amazon reserves the right, exercisable at its sole discretion, to modify these Guidelines and/or the approved Trademarks at any time and to take appropriate action against any use without permission or any use that does not conform to these Guidelines. If you have questions about these Guidelines, contact trademarks@amazon.com for assistance or write to us at the following address:

Amazon.com, Inc., Attention: Trademarks
PO Box 81226
Seattle, WA 98108-1226
Seller reports and data feeds

AWS Marketplace provides the following tools for collecting and analyzing information about your product sales:

- **Reports** (p. 153) that are automatically created and are available to all registered AWS Marketplace sellers.
- An **API** (p. 7) that enables you to access sections of those reports.
- **Data feeds** (p. 181) that provide additional customer information that you can use to identify customer information for transactions listed in the reports.

AWS Marketplace provides as much data as possible in reports and data feeds while adhering to the following:

- Amazon standards and tenets for protecting customer data.
- The terms and conditions that buyers accept when they buy a product on AWS Marketplace. As a seller, you are contractually bound to securely manage buyer data and to delete data upon buyer’s request.

**Seller reports**

AWS Marketplace provides reports that include information about product usage, buyers, billing, and payment information. Reports are available to all registered AWS Marketplace sellers.

Here are some key points about report generation:

- Reports are generated daily, weekly, or monthly, depending on the report.
- Reports are generated at 00:00 UTC and cover through 24:00 UTC of the previous day.
- Reports are generated as .csv files.
- You can configure Amazon SNS to notify you when data is delivered to your encrypted S3 bucket. After you configure notifications, AWS sends notifications to the email address that is associated with the AWS account that you registered with on AWS Marketplace.

For information on how to configure notifications, see Getting started with Amazon SNS in the Amazon Simple Notification Service Developer Guide.

To cancel getting notification emails, contact the AWS Marketplace Seller Operations team.

To learn about each report, you can download sample reports.

**Accessing reports**

AWS Marketplace provides two ways to access your reports:

- Using an API interface. The **AWS Marketplace Commerce Analytics Service** (p. 7) enables you to automatically access the data in your reports through an API interface. You can automate ingesting your information and download a portion of a report instead of the whole report. The service returns data asynchronously to a file in Amazon Simple Storage Service (Amazon S3) rather than directly as with a traditional API. The data is delivered in a machine-readable format so that you can import or incorporate the data into your systems.
- Using the reports dashboard in the **AWS Marketplace Management Portal**. This dashboard provides reports for previous reporting periods.
You can control access to reports by using AWS Identity and Access Management (IAM) permissions.

**Daily business report**

The daily business report helps you understand how AWS customers are using your products on a daily basis and the estimated revenue from that usage. You only receive this report if relevant information is available. If you don't receive this report and think that you should have received it, contact the AWS Marketplace Seller Operations team.

You can access this report at the AWS Marketplace Management Portal. If you are registered for the AWS Marketplace Commerce Analytics Service (p. 7), you can also access your reports using the AWS SDK.

You can use a unique identifier for each customer to identify customers over time and across reports. The identifier enables you to track customer usage patterns so that you can estimate customer spend, gain insights into free trial usage, and annual usage trends.

**Publication schedule**

This report is published daily at 00:00 UTC and covers from 00:00 UTC through 23:59 UTC of the previous day. Any exceptions to the schedule are noted at the introduction of the daily business report section.

**Topics**

- Section 1: Usage by instance type (p. 154)
- Section 2: Fees (p. 156)
- Section 3: Free trial conversions (p. 157)
- Section 4: New instances (p. 157)
- Section 5: New product subscribers (p. 158)
- Section 6: Canceled product subscribers (p. 159)

**Section 1: Usage by instance type**

This section lists data with a row for each instance type that the customer uses. For instance, when the customer uses a product on one instance type and the same product on a different instance type, the report includes a row for each of the two instance types.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Reference ID</td>
<td>A unique identifier that isn't the account ID. It helps track usage, revenue, and subscriptions by customers.</td>
</tr>
<tr>
<td>User's State</td>
<td>The billing address state that is associated with the account that is subscribed to the product.</td>
</tr>
<tr>
<td>User’s Country</td>
<td>The two-character country code that is associated with the account that is subscribed to the product. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Instance Type</td>
<td>The instance type associated with the product usage: for example, t2.micro.</td>
</tr>
<tr>
<td>Column name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Usage Units</td>
<td>The number of units of usage that the customer used during the reporting period.</td>
</tr>
<tr>
<td>Usage Unit Type</td>
<td>The unit of measurement that meters the customer's usage. For example, hours or days.</td>
</tr>
<tr>
<td>Offering Description</td>
<td>The description for product offering. For example, the product is offered for hourly usage, free trial usage, or annual usage.</td>
</tr>
<tr>
<td>Estimated Revenue</td>
<td>The estimated revenue from the product usage. The billing is finalized at the end of the month.</td>
</tr>
<tr>
<td>Currency</td>
<td>The currency of the transaction. For example, if the transaction is in US dollars, the entry is USD.</td>
</tr>
<tr>
<td>Offer ID</td>
<td>The identifier for the offer that the buyer signed.</td>
</tr>
<tr>
<td>Offer Visibility</td>
<td>Whether the offer is a public, private, or enterprise contract offer.</td>
</tr>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Country</td>
<td>The two-character country code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer State</td>
<td>The billing address state that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Customer City</td>
<td>The billing address city that is associated with the account that charges are billed to.</td>
</tr>
<tr>
<td>Customer Zip Code</td>
<td>The billing address zip code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to. For example, if the email address is <a href="mailto:liu-jie@example.com">liu-jie@example.com</a>, the entry is example.com.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Payer Address ID</td>
<td>A unique identifier that represents the customer's address.</td>
</tr>
</tbody>
</table>
Section 2: Fees

This section includes fee-based transactions that are associated with products: for example, annual, monthly, SaaS contracts product fees, and data product subscription fees. The data in this section covers the 24-hour period 72 hours before the time that the report is generated. For example, if the report is generated on May 24, the data covers the 24-hour period for May 21.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Reference ID</td>
<td>A unique identifier that isn't the account ID. It helps track usage, revenue, and subscriptions by customers.</td>
</tr>
<tr>
<td>User's State</td>
<td>The billing address state that is associated with the account that is subscribed to the product.</td>
</tr>
<tr>
<td>User's Country</td>
<td>The two-character country code that is associated with the account that is subscribed to the product. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Amount</td>
<td>The usage fee. If there is a refund, this value is negative. If this entry is for an AWS Marketplace SaaS contract, the amount represents the fee for the dimension, not the entire contract.</td>
</tr>
<tr>
<td>Currency</td>
<td>The currency of the transaction. For example, if the transaction is in US dollars, the entry is USD.</td>
</tr>
<tr>
<td>Fee Description</td>
<td>The reason for the fee: for example, monthly fee, annual fee, or refund.</td>
</tr>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Country</td>
<td>The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Customer State</td>
<td>The billing address state that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer City</td>
<td>The billing address city that is associated with the account that charges are billed to.</td>
</tr>
<tr>
<td>Customer Zip Code</td>
<td>The billing address zip code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to. For example, if the email address is <a href="mailto:liu-jie@example.com">liu-jie@example.com</a>, the entry is example.com.</td>
</tr>
<tr>
<td>Start Date</td>
<td>The start date for an AWS Marketplace SaaS contract or data product subscription.</td>
</tr>
</tbody>
</table>
Section 3: Free trial conversions

This section lists data for free trial starts, conversions and cancellations, and covers the previous 24-hour period.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier representing the product.</td>
</tr>
<tr>
<td>New Free Trials</td>
<td>The number of new free trials that are initiated in the reporting period.</td>
</tr>
<tr>
<td>Total Current Free Trials</td>
<td>The total number of active free trial subscriptions.</td>
</tr>
<tr>
<td>Converted Free Trials</td>
<td>The total number of subscriptions that moved from free trial to paid usage during the reporting period.</td>
</tr>
<tr>
<td>Non-Converted Free Trials</td>
<td>The total number of subscriptions that ended the free trial and didn't convert to paid usage.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
</tbody>
</table>

Section 4: New instances

This section lists data for new EC2 instance and instances types, and covers the previous 24-hour period.
<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Reference ID</td>
<td>A unique identifier that isn't the account ID. It helps track usage, revenue, and subscriptions by customers.</td>
</tr>
<tr>
<td>User's State</td>
<td>The billing address state that is associated with the account that is subscribed to the product.</td>
</tr>
<tr>
<td>User's Country</td>
<td>The two-character country code that is associated with the account that is subscribed to the product. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Type</td>
<td>The Amazon EC2 instance type.</td>
</tr>
<tr>
<td>Count</td>
<td>The number of EC2 instances.</td>
</tr>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Country</td>
<td>The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Customer State</td>
<td>The billing address state that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer City</td>
<td>The billing address city that is associated with the account that charges are billed to.</td>
</tr>
<tr>
<td>Customer Zip Code</td>
<td>The billing address zip code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to. For example, if the email address is <a href="mailto:liu-jie@example.com">liu-jie@example.com</a>, the entry is example.com.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Payer Address ID</td>
<td>A unique identifier that represents the customer's address.</td>
</tr>
</tbody>
</table>

**Section 5: New product subscribers**

This section lists data for new buyers, and covers the previous 24-hour period.
<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Reference ID</td>
<td>A unique identifier that isn't the account ID. It helps track usage, revenue, and subscriptions by customers.</td>
</tr>
<tr>
<td>User's State</td>
<td>The billing address state that is associated with the account that is subscribed to the product.</td>
</tr>
<tr>
<td>User's Country</td>
<td>The two-character country code that is associated with the account subscribed to the product. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Offer ID</td>
<td>The identifier for the offer the buyer signed.</td>
</tr>
<tr>
<td>Offer Visibility</td>
<td>Whether the offer is a public, private, or enterprise contract offer.</td>
</tr>
<tr>
<td>Customer Country</td>
<td>The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Customer State</td>
<td>The billing address state that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer City</td>
<td>The billing address city that is associated with the account that charges are billed to.</td>
</tr>
<tr>
<td>Customer Zip Code</td>
<td>The billing address zip code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to. For example, if the email address is <a href="mailto:liu-jie@example.com">liu-jie@example.com</a>, the entry is example.com.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Payer Address ID</td>
<td>A unique identifier that represents the customer's address.</td>
</tr>
</tbody>
</table>

**Section 6: Canceled product subscribers**

This section lists data for buyer cancellations, and covers the previous 24-hour period.
<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Reference ID</td>
<td>A unique identifier that isn't the account ID. It helps track usage, revenue, and subscriptions by customers.</td>
</tr>
<tr>
<td>User's State</td>
<td>The billing address state that is associated with the account that is subscribed to the product.</td>
</tr>
<tr>
<td>User's Country</td>
<td>The two-character country code that is associated with the account that is subscribed to the product. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Subscribed Date</td>
<td>The date when the subscription started.</td>
</tr>
<tr>
<td>Offer ID</td>
<td>The identifier for the offer that the buyer signed.</td>
</tr>
<tr>
<td>Offer Visibility</td>
<td>Whether the offer is a public, private, or enterprise contract offer.</td>
</tr>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Country</td>
<td>The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Customer State</td>
<td>The billing address state that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer City</td>
<td>The billing address city that is associated with the account that charges are billed to.</td>
</tr>
<tr>
<td>Customer Zip Code</td>
<td>The billing address zip code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to. For example, if the email address is <a href="mailto:liu-jie@example.com">liu-jie@example.com</a>, the entry is example.com.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Payer Address ID</td>
<td>A unique identifier that represents the customer's address.</td>
</tr>
</tbody>
</table>
Daily customer subscriber report

This report lists data for customers who purchased your products. This report doesn't specify current or past usage, only that a customer is subscribed to your product. You only receive this report if relevant information is available. If you don't receive this report and think that you should have, contact the AWS Marketplace Seller Operations team.

You can access this report at the AWS Marketplace Management Portal. If you are registered for the section called “AWS Marketplace Commerce Analytics Service” (p. 7), you can also access your reports using the AWS SDK.

The report has two sections: one for hourly and monthly subscriptions and one for annual subscriptions. The report includes the list of AWS account IDs for all customers who are subscribed to your products.

Publication schedule

This report is published daily at 00:00 UTC and covers from 00:00 UTC through 23:59 UTC of the previous day.

Topics

- Section 1: Hourly and monthly subscriptions (p. 161)
- Section 2: Variable length subscriptions (p. 162)

Section 1: Hourly and monthly subscriptions

This section lists data for all usage-based subscriptions as of the previous day at 23:59:59 UTC.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer AWS Account Number</td>
<td>The account that is subscribed to the product.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Id</td>
<td>A unique identifier for the software product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the software product.</td>
</tr>
<tr>
<td>Subscription Start Date</td>
<td>The start date for the subscription, formatted as YYYY-MM-DD.</td>
</tr>
<tr>
<td>Offer ID</td>
<td>The identifier for the offer that the buyer signed.</td>
</tr>
<tr>
<td>Offer Visibility</td>
<td>Whether the offer is a public, private, or enterprise contract offer.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Reseller account ID</td>
<td>The unique identifier for the consulting partner reseller.</td>
</tr>
</tbody>
</table>
Section 2: Variable length subscriptions

This section lists data for all fee-based subscriptions as of the previous day at 23:59:59 UTC.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the account that is subscribed to the product.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Id</td>
<td>The unique identifier for the software product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>A unique identifier for the software product. This information is also available as part of the Amazon EC2 instance metadata.</td>
</tr>
<tr>
<td>Subscription Id</td>
<td>The ID for the subscription.</td>
</tr>
<tr>
<td>Subscription Quantity</td>
<td>The total number of licenses that the customer purchased.</td>
</tr>
<tr>
<td>Subscription Type</td>
<td>The type of subscription.</td>
</tr>
<tr>
<td>Subscription Intent</td>
<td>Whether this offer is an upgrade or renewal of an earlier offer.</td>
</tr>
<tr>
<td>Offer ID</td>
<td>The identifier for the offer that the buyer signed.</td>
</tr>
<tr>
<td>Subscription Start Date</td>
<td>The date when the customer subscribed to the product, formatted as YYYY-MM-DD.</td>
</tr>
<tr>
<td>Previous Offer ID</td>
<td>The ID of the offer that preceded the upgrade or renewal offer, if one exists.</td>
</tr>
<tr>
<td>Offer Visibility</td>
<td>Whether the offer is a public, private, or enterprise contract offer.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Reseller account ID</td>
<td>The unique identifier for the consulting partner reseller.</td>
</tr>
<tr>
<td>Reseller account name</td>
<td>The name of the consulting partner reseller.</td>
</tr>
</tbody>
</table>
Disbursement report

The disbursement report provides information about funds that we collected and disbursed to your bank accounts since the previous disbursement. Disbursements can include customer payments or refunds for a subscription to your product, and some taxes collected or refunded to the customer. You don’t receive disbursement of funds until the funds are collected from the customer. Different customers have different payment terms with AWS, so some of the funds in each of the uncollected age categories might not be due from the customer.

Refunds appear as negative amounts because the money is returned to your customer after you authorize a refund.

This report is available on the AWS Marketplace Management Portal under the Reports tab. If you’re enrolled in the AWS Marketplace commerce analytics service, you can use API calls to pull down sections of this report. For more information, see the section called "AWS Marketplace Commerce Analytics Service" (p. 7).

Publication schedule

This report is published 3-5 days after a disbursement has been initiated to transfer funds to your bank. In general, this is a report for sellers who receive disbursements on a monthly cadence. If there is no disbursement initiated, no disbursement report is generated.

Topics

- Section 1: Disbursed amount by product (p. 163)
- Section 2: Disbursed amount by customer geography (p. 165)
- Section 3: Disbursed amount by instance hours (p. 166)
- Section 4: Age of uncollected funds (p. 166)
- Section 5: Age of disbursed funds (p. 167)
- Section 6: Age of past due funds (p. 167)
- Section 7: Uncollected funds breakdown (p. 167)

Section 1: Disbursed amount by product

This section lists data for disbursements by product.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>SellerRev</td>
<td>The amount that is billed to the customer for the usage or fees of the product.</td>
</tr>
<tr>
<td>AWSRefFee</td>
<td>The amount of the AWS Marketplace fee.</td>
</tr>
<tr>
<td>SellerRevRefund</td>
<td>The amount of the subscription cost that is refunded to customers if any refunds were processed during the data coverage period.</td>
</tr>
<tr>
<td>AWSRefFeeRefund</td>
<td>The amount of the AWS Marketplace fee that is refunded if any refunds were processed during the data coverage period.</td>
</tr>
<tr>
<td>Column name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SellerRevCredit</td>
<td>The AWS credits that AWS Marketplace placed on the customer's account.</td>
</tr>
<tr>
<td>AWSRefFeeCredit</td>
<td>The AWS credits that AWS Marketplace placed on your account.</td>
</tr>
<tr>
<td>Net Amount</td>
<td>The total funds that we disbursed to you. This column is equal to the SellerRev column minus the AWSRefFee column. When a refund is given to a customer, this column is a negative number equal to the SellerRevRefund column minus the AWSRefFeeRefund column.</td>
</tr>
<tr>
<td>Transaction Reference ID</td>
<td>A unique identifier for the transaction that helps you correlate transactions across AWS Marketplace reports.</td>
</tr>
<tr>
<td>SellerUSSalesTax</td>
<td>The total amount of US sales and use tax that is billed for this transaction.</td>
</tr>
<tr>
<td>SellerUSSalesTaxRefund</td>
<td>The total amount of US sales and use tax that is refunded for this transaction if a refund was processed.</td>
</tr>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Country</td>
<td>The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Customer State</td>
<td>The billing address state that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer City</td>
<td>The billing address city that is associated with the account that charges are billed to.</td>
</tr>
<tr>
<td>Customer Zip Code</td>
<td>The billing address postal code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to. For example, if the email address is <a href="mailto:liu-jie@example.com">liu-jie@example.com</a>, the entry is example.com.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
<tr>
<td>Launch Type Description</td>
<td>The type of instance that the customer launched. This is Amazon EC2 or AWS Fargate.</td>
</tr>
<tr>
<td>Container Hours</td>
<td>The aggregate partial hours per Region by launch type.</td>
</tr>
</tbody>
</table>
Section 2: Disbursed amount by customer geography

This section lists data for disbursements by the customer's geographic location.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Settlement ID</td>
<td>The unique identifier of the disbursement.</td>
</tr>
<tr>
<td>Settlement Period Start Date</td>
<td>The starting date and time of the disbursement period.</td>
</tr>
<tr>
<td>Settlement Period End Date</td>
<td>The ending date and time of the disbursement period.</td>
</tr>
<tr>
<td>Deposit Date</td>
<td>The date and time when the disbursement occurred.</td>
</tr>
<tr>
<td>Disbursed Amount</td>
<td>The total amount of the disbursement.</td>
</tr>
<tr>
<td>Country Code</td>
<td>The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>State or Region</td>
<td>The billing address state that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>City</td>
<td>The billing address city that is associated with the account that charges are billed to.</td>
</tr>
<tr>
<td>Postal Code</td>
<td>The billing address postal code that is associated with the account that the software charges are billed to.</td>
</tr>
<tr>
<td>Net Amount by Tax Location</td>
<td>The total funds that are disbursed to the seller by tax location, less AWS Marketplace fees, refunds, and US sales and use tax.</td>
</tr>
<tr>
<td>Gross Amount by Tax Location</td>
<td>The total funds that are disbursed to the seller by tax location.</td>
</tr>
<tr>
<td>Seller U.S. Sales Tax</td>
<td>The total amount of US sales and use tax that is billed for this transaction on behalf of the Seller. (That is, related records in US Sales and Tax reports show &quot;tax liable party&quot; == &quot;SELLER&quot;).</td>
</tr>
<tr>
<td>Seller U.S. Sales Tax Refund</td>
<td>The total amount of US sales and use tax that is refunded for this transaction if a refund was</td>
</tr>
</tbody>
</table>
Section 3: Disbursed amount by instance hours

This section lists data for disbursements by Amazon EC2 instance hours.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Usage Type Description</td>
<td>The description of the usage, including offer type, Region, and instance type.</td>
</tr>
<tr>
<td>Rate</td>
<td>The rate per hour for the offer type, Region, and instance type.</td>
</tr>
<tr>
<td>User Count</td>
<td>The number of unique customers using the offer type, Region, and instance type.</td>
</tr>
<tr>
<td>Instance Hours</td>
<td>The number of hours that the instance consumed for the offer type, Region, and instance type.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
</tbody>
</table>

Section 4: Age of uncollected funds

This section lists data for uncollected funds, organized by the age. Uncollected funds might include amounts that aren't due yet.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncollected (&lt; 31 days pending)</td>
<td>The total of funds billed but not collected for less than 31 days.</td>
</tr>
<tr>
<td>Uncollected (31–60 days pending)</td>
<td>The total of funds billed but not collected for between 31–60 days.</td>
</tr>
<tr>
<td>Uncollected (61–90 days pending)</td>
<td>The total of funds billed but not collected for between 61–90 days.</td>
</tr>
<tr>
<td>Uncollected (91–120 days pending)</td>
<td>The total of funds billed but not collected for between 91–120 days.</td>
</tr>
<tr>
<td>Uncollected (&gt; 120 days pending)</td>
<td>The total of funds billed but not collected for more than 120 days.</td>
</tr>
<tr>
<td>Uncollected (overall)</td>
<td>The total of all funds billed but not collected.</td>
</tr>
</tbody>
</table>
Section 5: Age of disbursed funds

This section lists data for collected funds since the previous disbursement.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collected (&lt; 31 days pending)</td>
<td>The total of funds collected that were billed in the 0–31 day range.</td>
</tr>
<tr>
<td>Collected (31–60 days pending)</td>
<td>The total of funds collected that were billed in the 31–60 day range.</td>
</tr>
<tr>
<td>Collected (61–90 days pending)</td>
<td>The total of funds collected that were billed in the 61–90 days range.</td>
</tr>
<tr>
<td>Collected (91–120 days pending)</td>
<td>The total of funds collected that were billed in the 91–120 days range.</td>
</tr>
<tr>
<td>Collected (&gt; 120 days pending)</td>
<td>The total of funds collected that were billed in the greater than 120 days range.</td>
</tr>
<tr>
<td>Collected (overall)</td>
<td>The total of all collected funds.</td>
</tr>
</tbody>
</table>

Section 6: Age of past due funds

This section lists data for funds that have been accrued and are payable by the customer, but have not been paid in accordance with the customer's agreement with AWS.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Past Due (&lt; 31 days)</td>
<td>The total of funds that have accrued in the last 0–31 days and are due but that the customer hasn't paid.</td>
</tr>
<tr>
<td>Past Due (31–60 days)</td>
<td>The total of funds that have accrued in the last 31–60 days and are due but that the customer hasn't paid.</td>
</tr>
<tr>
<td>Past Due (61–90 days)</td>
<td>The total of funds that have accrued in the last 61–90 days that are due but that the customer hasn't paid.</td>
</tr>
<tr>
<td>Past Due (91–120 days)</td>
<td>The total of funds that have accrued in the last 91–120 days and are due but that the customer hasn't paid.</td>
</tr>
<tr>
<td>Past Due (&gt; 120 days)</td>
<td>The total of funds that have accrued in the last 121 or more days and are due but that the customer hasn't paid.</td>
</tr>
<tr>
<td>Past Due (overall)</td>
<td>The total of funds that have accrued and are due but that the customer hasn't paid.</td>
</tr>
</tbody>
</table>

Section 7: Uncollected funds breakdown

This section lists all uncollected funds, sorted by the payment due date.
<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payer AWS Account Number</td>
<td>The account that the software charges are billed to.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Gross Revenue</td>
<td>The amount that is billed for using the product or the fees for using the product.</td>
</tr>
<tr>
<td>AWS Revenue Share</td>
<td>The AWS fee amount that is deducted from the billed amount at settlement time.</td>
</tr>
<tr>
<td>Gross Refunds</td>
<td>The total amount of any refunds for the transaction.</td>
</tr>
<tr>
<td>AWS Refunds Share</td>
<td>The portion of the AWS fee that is refunded for the transaction.</td>
</tr>
<tr>
<td>Net Revenue</td>
<td>The net amount that is billed for this transaction, minus AWS fees, refunds, and US sales and use tax.</td>
</tr>
<tr>
<td>Currency</td>
<td>The currency of the transaction. For example, if the transaction is in US dollars, the entry is USD.</td>
</tr>
<tr>
<td>AR Period</td>
<td>The month and year of the transaction, in the format of YYYY-MM.</td>
</tr>
<tr>
<td>Transaction Reference ID</td>
<td>A unique identifier that represents the transaction, which you can use to correlate transactions across AWS Marketplace reports.</td>
</tr>
<tr>
<td>Opportunity Name</td>
<td>The unique identifier for a registered opportunity.</td>
</tr>
<tr>
<td>Opportunity Description</td>
<td>Any metadata in the registered opportunity.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier of the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Payer Address ID</td>
<td>A unique identifier that represents the customer's address.</td>
</tr>
<tr>
<td>Payment Due date</td>
<td>The payment due date in the format of YYYY-MM-DD.</td>
</tr>
</tbody>
</table>

**Monthly billed revenue report**

The monthly billed revenue report provides you authoritative information about billed revenue every month for accounting and other financial reporting purposes. This report shows the total amounts that AWS bills to customers for hourly, annual, or monthly usage of your products. The report has
four sections: billed amounts for hourly usage and monthly fees, variable-length subscriptions, field demonstration usage, and flexible payments.

**Important**
The amounts in this report reflect only revenue that we billed to customers, not amounts that we collected.

This report is available on the AWS Marketplace Management Portal under the **Reports** tab. If you’re enrolled in the AWS Marketplace commerce analytics service, you can use API calls to pull down sections of this report. For more information, see the section called “AWS Marketplace Commerce Analytics Service” (p. 7).

**Publication schedule**

This report is published monthly on the fifteenth day of each month at 00:00 UTC. The report covers the previous calendar month from the first day of the month at 00:00 UTC through the last day of the month at 23:59 UTC. For example, the report that is published on May 15 covers from April 1 at 00:00 UTC through April 30 at 23:59 UTC.

**Topics**

- Section 1: Billing and revenue data (p. 169)
- Section 2: Variable length subscriptions (p. 171)
- Section 3: AWS field demonstration usage (p. 173)
- Section 4: Contracts with flexible payment schedule (p. 173)

**Section 1: Billing and revenue data**

This section lists data for usage billing, refunds, fees, and US sales and use tax that is collected.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Reference ID</td>
<td>A unique identifier that isn't the account ID. It helps track usage, revenue, and subscriptions by customers.</td>
</tr>
<tr>
<td>Country</td>
<td>The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>State</td>
<td>The billing address state that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>City</td>
<td>The billing address city that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Zip Code</td>
<td>The billing address postal code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Customer Billed Amount</td>
<td>The amount that is billed to the customer for the usage or monthly fees of the product.</td>
</tr>
<tr>
<td>AWS Listing Fee</td>
<td>The AWS Marketplace fee amount to be deducted from the billed amount.</td>
</tr>
<tr>
<td>Column name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Refunds Amount</td>
<td>The total amount of the subscription cost refunded to customers if any refunds were processed during the data coverage period.</td>
</tr>
<tr>
<td>AWS Fee Refund</td>
<td>The portion of the AWS Marketplace fee refunded if any refunds were processed during the data coverage period.</td>
</tr>
<tr>
<td>Cost</td>
<td>The cost of goods to a reseller: for example, what a reseller pays you when they sell your product.</td>
</tr>
<tr>
<td>Partner Revenue Amount</td>
<td>The total amount billed for the transaction, net of AWS Marketplace fees, refunds, and US sales and use tax.</td>
</tr>
<tr>
<td>Currency</td>
<td>The currency of the transaction. For example, if the transaction is in US dollars, the entry is USD.</td>
</tr>
<tr>
<td>Transaction Reference ID</td>
<td>A unique identifier for the transaction that helps you correlate transactions across AWS Marketplace reports.</td>
</tr>
<tr>
<td>U.S. Sales Tax Customer Billed Amount</td>
<td>The total amount of US sales and use tax that is billed for this transaction on behalf of the Seller. (That is, related records in US Sales and Tax reports show “tax liable party” == “SELLER”.)</td>
</tr>
<tr>
<td>U.S. Sales Tax Refunds Amount</td>
<td>The total amount of US sales and use tax that is refunded for this transaction if a refund was processed, when such taxes were collected on behalf of the Seller. (That is, related records in US Sales and Tax reports show “tax liable party” == “SELLER”.)</td>
</tr>
<tr>
<td>Offer ID</td>
<td>The identifier for the offer that the buyer signed.</td>
</tr>
<tr>
<td>Offer Visibility</td>
<td>Whether the offer is a public, private, or enterprise contract offer.</td>
</tr>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Opportunity Name</td>
<td>The unique identifier for a registered opportunity.</td>
</tr>
<tr>
<td>Opportunity Description</td>
<td>The metadata for the registered opportunity.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
</tbody>
</table>
### Column name | Description
---|---
Payer Reference ID | A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.
Payer Address ID | A unique identifier that represents the customer's address.

### Section 2: Variable length subscriptions

This section lists data for fee-based charges.

| Column name          | Description                                                                 |
---|---|
Customer Reference ID | A unique identifier that isn't the account ID. It helps track usage, revenue, and subscriptions by customers. |
Country               | The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard. |
State                 | The billing address state that is associated with the account that the charges are billed to. |
City                  | The billing address city that is associated with the account that the charges are billed to. |
Zip Code               | The billing address zip code that is associated with the account that the charges are billed to. |
Product Title          | The title of the product. |
Product Code           | The unique identifier for the product. |
Subscription Quantity   | The number of total licenses that is specified as part of the variable-length subscription purchase. |
Subscription Start Date | The start date of the variable-length subscription purchase. |
Subscription End Date   | The end date of the variable-length subscription purchase. |
Subscription Instance Type | The instance type that is associated with the variable-length subscription purchase. |
Customer Billed Amount | The amount that is billed for the usage, monthly fees, or both. |
AWS Listing Fee         | The AWS Marketplace fee amount that is deducted from the billed amount. |
<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refunds Amount</td>
<td>The total amount refunded to customers if any refunds were processed during the data coverage period.</td>
</tr>
<tr>
<td>AWS Fee Refund</td>
<td>The portion of the AWS Marketplace fee refunded if any refunds were processed during the data coverage period.</td>
</tr>
<tr>
<td>Cost</td>
<td>The cost of goods to a reseller: for example, what a reseller pays you when they sell your product.</td>
</tr>
<tr>
<td>Partner Revenue Amount</td>
<td>The total amount that is billed for this transaction, net of AWS Marketplace fees, refunds, and US sales and use tax.</td>
</tr>
<tr>
<td>Currency</td>
<td>The currency of the transaction. For example, if the transaction is in US dollars, the entry is USD.</td>
</tr>
<tr>
<td>Transaction Reference ID</td>
<td>A unique identifier for the transaction that helps you correlate transactions across AWS Marketplace reports.</td>
</tr>
<tr>
<td>U.S. Sales Tax Customer Billed Amount</td>
<td>The total amount of US sales and use tax that is billed for this transaction on behalf of the Seller. (That is, related records in US Sales and Tax reports show “tax liable party” == “SELLER”.)</td>
</tr>
<tr>
<td>U.S. Sales Tax Refunds Amount</td>
<td>The total amount of US sales and use tax that is refunded for this transaction if a refund was processed, when such taxes were collected on behalf of the Seller. (That is, related records in US Sales and Tax reports show “tax liable party” == “SELLER”.)</td>
</tr>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to. For example, if the email address is <a href="mailto:liu-jie@example.com">liu-jie@example.com</a>, the entry is example.com.</td>
</tr>
<tr>
<td>Offer ID</td>
<td>The identifier for the offer that the buyer signed.</td>
</tr>
<tr>
<td>Offer Visibility</td>
<td>Whether the offer is a public, private, or enterprise contract offer.</td>
</tr>
<tr>
<td>Contract Start Date</td>
<td>The start date for an AWS Marketplace SaaS contract.</td>
</tr>
<tr>
<td>Contract End Date</td>
<td>The end date for an AWS Marketplace SaaS contract.</td>
</tr>
<tr>
<td>Opportunity Name</td>
<td>The unique identifier for a registered opportunity.</td>
</tr>
<tr>
<td>Opportunity Description</td>
<td>The metadata for the registered opportunity.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Column name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier for the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Payer Address ID</td>
<td>A unique identifier that represents the customer's address.</td>
</tr>
</tbody>
</table>

**Section 3: AWS field demonstration usage**

The section lists data for AWS field demonstration usage (p. 19) of your product. You can configure your product to allow us to demonstrate your product to potential customers. Any usage from the demonstrations is listed here.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Instance Type</td>
<td>The Amazon EC2 instance type that is associated with the field demonstration.</td>
</tr>
<tr>
<td>Usage Units</td>
<td>The number of units of usage that is associated with the product.</td>
</tr>
<tr>
<td>Usage Unit Types</td>
<td>The usage units that are associated with the usage unit count: for example, hours.</td>
</tr>
</tbody>
</table>

**Section 4: Contracts with flexible payment schedule**

This section lists data for all contracts that you created with a flexible payment schedule in the previous reporting period.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer AWS Account Number</td>
<td>The ID of the payer account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer Country</td>
<td>The two-character country code that is associated with the payer account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Customer State</td>
<td>The billing address state that is associated with the payer account that the charges are billed to.</td>
</tr>
<tr>
<td>Customer City</td>
<td>The billing address city that is associated with the payer account that charges are billed to.</td>
</tr>
</tbody>
</table>
### Column name | Description
--- | ---
Customer ZIP Code | The billing address zip code that is associated with the payer account that the charges are billed to.
Customer Email Domain | The email domain that is associated with the payer account that the charges are billed to. For example, if the email address is liu-jie@example.com, the entry is example.com.
User Reference ID | The account of the payer account that the charges are billed to.
User AWS Account Number | The ID of the account that subscribed to the product.
Product ID | The unique identifier for the product.
Product Title | The title of the product.
Product Type | The type of product.
AWS Marketplace Offer ID | The identifier for the offer that the buyer signed.
Contract Create Date | The contract creation date, which is the date that an account subscribes to the offer.
Contract Expiration Date | The date when the contract expires.
Total Contract Value (USD) | The total value of the contract in USD.
# of Payments | The number of payments that are scheduled for the contract.
Invoice Date | The date the invoice is created.
Invoice Amount (USD) | The amount that is billed on the invoice in USD.
Payer Reference ID | A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.

## Sales compensation report

The report lists monthly billed revenue with additional customer information that isn't in the standard the section called "Monthly billed revenue report" (p. 168). The report lists the total amounts that AWS bills to customers for hourly, annual, or monthly usage of your product.

**Important**
The amounts in this report reflect only revenue that is billed to customers, not amounts that are collected from customers.

The sales compensation report and the information that is shared with you as part of this program constitute Amazon's Confidential Information under our nondisclosure agreement with you or, if no such agreement exists, the Terms and Conditions for AWS Marketplace sellers. You can use this information only for compensating your sales representatives by mapping AWS Marketplace revenue to the representatives by company name, geography, and AWS account ID. You can share this information
with employees who need to know it to understand the source of commissions that is payable to them. Your use and sharing of such information must comply with the obligations in our nondisclosure agreement with you and the terms and conditions for AWS Marketplace sellers, including, without limitation, Section 3.8 of the Terms and Conditions for AWS Marketplace sellers.

**Publication schedule**

This report is published monthly, on the fifteenth day of each month at 00:00 UTC. The report covers the previous calendar month from 00:00 UTC through 23:59 UTC of the last calendar day of the month. For example, the report published on May 15 covers from April 1 at 00:00 UTC through April 30 at 23:59 UTC.

**Billed revenue**

The billed revenue section of this report includes usage and fee-based charges from the previous calendar month. The following are the column names and descriptions.

**Note**

In this report, *listing fee* is the percentage of transaction proceeds (except for those from resale by authorized resellers of authorized resale products) determined in accordance with the tiered listing fee.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer AWS Account Number</td>
<td>The account that the charges are billed to.</td>
</tr>
<tr>
<td>Country</td>
<td>The two-character country code that is associated with the account that the charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>State</td>
<td>The billing address state that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>City</td>
<td>The billing address city that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Zip Code</td>
<td>The billing address zip code that is associated with the account that the charges are billed to.</td>
</tr>
<tr>
<td>Email Domain</td>
<td>The email domain that is associated with the account that the charges are billed to. For example, if the email address is <a href="mailto:liu-jie@example.com">liu-jie@example.com</a>, the entry is example.com.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Product Title</td>
<td>The title of the product.</td>
</tr>
<tr>
<td>Gross Revenue</td>
<td>The amount that is billed for using the product or the monthly fees for using the product.</td>
</tr>
<tr>
<td>AWS Revenue Share</td>
<td>The AWS fee amount that is deducted from the billed amount at settlement time. It appears in the section called “Disbursement report” (p. 163).</td>
</tr>
<tr>
<td>Gross Refunds</td>
<td>The total amount of any refunds for the transaction.</td>
</tr>
<tr>
<td>Column name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AWS Refunds Share</td>
<td>The portion of the AWS fee that is refunded for the transaction.</td>
</tr>
<tr>
<td>Net Revenue</td>
<td>The net amount that is billed for this transaction, minus AWS fees, refunds, and US sales and use tax.</td>
</tr>
<tr>
<td>Currency</td>
<td>The currency of the transaction. For example, if the transaction is in US dollars, the entry is USD.</td>
</tr>
<tr>
<td>AR Period</td>
<td>The month and year of the transaction, in the format of YYYY-MM.</td>
</tr>
<tr>
<td>Transaction Reference ID</td>
<td>A unique identifier that represents the transaction, which you can use to correlate transactions across AWS Marketplace reports.</td>
</tr>
<tr>
<td>Opportunity Name</td>
<td>The unique identifier for a registered opportunity.</td>
</tr>
<tr>
<td>Opportunity Description</td>
<td>Any metadata in the registered opportunity.</td>
</tr>
<tr>
<td>Solution Title</td>
<td>The name of the solution.</td>
</tr>
<tr>
<td>Solution ID</td>
<td>The unique identifier of the solution.</td>
</tr>
<tr>
<td>Payer Reference ID</td>
<td>A unique identifier that isn't the account ID. It's associated with the account that fees are billed to. It helps with tracking usage, revenue, and subscriptions by customers across all of the AWS Marketplace financial reports.</td>
</tr>
<tr>
<td>Payer Address ID</td>
<td>A unique identifier that represents the customer's address.</td>
</tr>
</tbody>
</table>

**US sales and use tax report**

This monthly report provides sellers with information about US sales and use tax that AWS collects from sales and use transactions in AWS Marketplace. The report includes both products that sellers enroll in the AWS Marketplace US sales tax collection service and products that AWS is required to collect and remit tax on.

For sales of products enrolled in the tax calculation service, the report includes calculated US sales and use tax for products with a product tax code. Any products without a product tax code appear in this report with a tax value of $0.00 USD. For sales of products that are not eligible for the tax calculation service because of enacted marketplace facilitator rules, you will see amounts that AWS has collected and remitted as AWS, based on our internal tax decisions. For more information, see AWS Marketplace Sellers & Tax Collection on Amazon Web Services Tax Help.

To map transactions between the disbursement report and this report, use the **Transaction Reference ID**.

This report is available on the AWS Marketplace Management Portal under the **Reports** tab. If you're enrolled in the AWS Marketplace commerce analytics service, you can use API calls to pull down sections of this report. For more information, see the section called "AWS Marketplace Commerce Analytics Service" (p. 7).
**Publication schedule**

This report is published monthly on the fifteenth day of each month at 00:00 UTC. The report covers the previous calendar month from the first day of the month at 00:00 UTC through the last day of the month at 23:59 UTC. For example, the report that is published on May 15 covers from April 1 at 00:00 UTC through April 30 at 23:59 UTC.

**US sales and use tax records**

This section lists data for US sales tax amounts that result from software charges.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Item ID</td>
<td>A unique identifier for a line item. Refund transactions have the same line item ID as their forward tax transactions.</td>
</tr>
<tr>
<td>Customer Bill ID</td>
<td>The unique identifier for a customer bill.</td>
</tr>
<tr>
<td>Product Name</td>
<td>The name of the product purchased.</td>
</tr>
<tr>
<td>Product Code</td>
<td>The unique identifier for the product.</td>
</tr>
<tr>
<td>Product Tax Code</td>
<td>A standard code to identify the tax properties for a product. You choose the properties when you create or modify the product.</td>
</tr>
<tr>
<td>Seller ID</td>
<td>A unique identifier for the seller of record of the transaction.</td>
</tr>
<tr>
<td>Seller Name</td>
<td>The legal name of the seller.</td>
</tr>
<tr>
<td>Transaction Date</td>
<td>The date of the transaction.</td>
</tr>
<tr>
<td>Total Adjusted Price</td>
<td>The final price for the transaction.</td>
</tr>
<tr>
<td>Total Tax</td>
<td>The total tax that is charged for the transaction.</td>
</tr>
<tr>
<td>Base Currency Code</td>
<td>The base currency code for all AWS Marketplace transactions. This entry is always USD.</td>
</tr>
<tr>
<td>Bill to City</td>
<td>The billing address city that is associated with the payer account that we bill software charges to.</td>
</tr>
<tr>
<td>Bill to State</td>
<td>The billing address zip code that is associated with the payer account that the software charges are billed to.</td>
</tr>
<tr>
<td>Bill to Postal Code</td>
<td>The billing address postal code that is associated with the payer account that the software charges are billed to.</td>
</tr>
<tr>
<td>Bill to Country</td>
<td>The two-character country code that is associated with the payer account that the software charges are billed to. This report uses ISO 3166-1 alpha-2 standard.</td>
</tr>
<tr>
<td>Transaction Type Code</td>
<td>The type code of the transaction. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• AWS: A forward tax transaction</td>
</tr>
<tr>
<td>Column name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Refund</strong></td>
<td>• <strong>REFUND</strong>: A full or partial refund</td>
</tr>
<tr>
<td></td>
<td>• <strong>TAXONLYREFUND</strong>: A tax-only refund</td>
</tr>
<tr>
<td></td>
<td>Refund transactions share the line item ID with their original forward transactions.</td>
</tr>
<tr>
<td><strong>Display Price Taxability Type</strong></td>
<td>The taxability type for the price that appears to customers. All AWS Marketplace offerings are exclusive.</td>
</tr>
<tr>
<td><strong>Tax Location Code Taxed Jurisdiction</strong></td>
<td>The vertex geocode that is associated with the taxed location.</td>
</tr>
<tr>
<td><strong>Tax Type Code</strong></td>
<td>The type of tax that is applied to the transaction. The possible values are None, Sales, and SellerUse.</td>
</tr>
<tr>
<td><strong>Jurisdiction Level</strong></td>
<td>The jurisdiction level of the address that is used for tax location. The possible values are State, County, City, and District.</td>
</tr>
<tr>
<td><strong>Taxed Jurisdiction</strong></td>
<td>The name of the taxed jurisdiction.</td>
</tr>
<tr>
<td><strong>Taxable Sale Amount</strong></td>
<td>The amount of the transaction that is taxable, by jurisdiction level.</td>
</tr>
<tr>
<td><strong>Nontaxable Sale Amount</strong></td>
<td>The amount of the transaction that is nontaxable, by jurisdiction level.</td>
</tr>
<tr>
<td><strong>Tax Amount</strong></td>
<td>The tax that is charged at the jurisdiction level.</td>
</tr>
<tr>
<td><strong>Tax Jurisdiction Tax Rate</strong></td>
<td>The tax rate that is applied at the jurisdiction level.</td>
</tr>
<tr>
<td><strong>Tax Calculation Reason Code</strong></td>
<td>Whether the transaction is taxable, not taxable, exempt, or zero-rated, organized by the jurisdiction level.</td>
</tr>
<tr>
<td><strong>Date Used For Tax Calculation</strong></td>
<td>The date that is used for calculating tax on the transaction.</td>
</tr>
<tr>
<td><strong>Customer Exemption Certificate ID</strong></td>
<td>The certificate ID of the exemption certificate.</td>
</tr>
<tr>
<td><strong>Customer Exemption Certificate ID Domain</strong></td>
<td>Where the certificate is being stored in Amazon systems.</td>
</tr>
<tr>
<td><strong>Customer Exemption Certificate Level</strong></td>
<td>The jurisdiction level that supplied the exemption.</td>
</tr>
<tr>
<td><strong>Customer Exemption Code</strong></td>
<td>The code that specifies the exemption: for example, RESALE.</td>
</tr>
<tr>
<td><strong>Customer Exemption Domain</strong></td>
<td>The Amazon system that is used to capture the customer exemption information, if information is available.</td>
</tr>
<tr>
<td><strong>Customer Reference ID</strong></td>
<td>A unique identifier that isn't the account ID. It helps track usage, revenue and subscriptions by customers.</td>
</tr>
</tbody>
</table>
Daily ref tag

This report lists data from the Marketing tab of the AWS Marketplace Management Portal and provides insight into clicks and conversions for ref tag links that customers use to get to your AWS Marketplace product. This report is not automatically emailed to you, and you can't use the AWS Commerce Analytics Service to pull the data in the report.

For more information about how to use this data, watch Getting Started with AWS Marketplace Marketing Analytics.

Publication schedule

This report is published daily at 00:00 UTC and covers from 00:00 UTC through 23:59 UTC of the previous day.

Clicks and conversions

The following table explains the ref tags that are used with your products and the number of clicks, conversions, estimated usage, and estimated revenue that are associated with them.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATETIME_DAY</td>
<td>The day that we received the ref tag link.</td>
</tr>
<tr>
<td>TITLE</td>
<td>The name of the product.</td>
</tr>
<tr>
<td>PRODUCT_CODE</td>
<td>A unique identifier for the product that is associated with billing. It's also available in Amazon EC2 instance metadata.</td>
</tr>
<tr>
<td>ASIN</td>
<td>A unique identifier for your product. It's used in your URL.</td>
</tr>
<tr>
<td>REFTAG</td>
<td>The name of the ref tag.</td>
</tr>
</tbody>
</table>
**Weekly ref tag**

This report lists data from the Marketing tab of the AWS Marketplace Management Portal. It provides insight into clicks and conversions for ref tag links that customers use to get to your AWS Marketplace product page. The report includes only ref tags that contain _ptnr_ or SEM/Online ref tags that start with ads_.

For more information about how to use this data, watch Getting Started with AWS Marketplace Marketing Analytics. For information about setting up ref tags, see Suggested ref tags for demand generation to product detail pages.

**Publication schedule**

This report is published weekly at 00:00 UTC and covers from 00:00 UTC through 23:59 UTC of the previous calendar week.

**Clicks and conversions**

The following table explains the ref tags that are used with your products and the number of clicks and conversions that are associated with them.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLICKS</td>
<td>The number of visits to your page with date, ref tag, and Amazon Standard Identification Number (ASIN) grouping.</td>
</tr>
<tr>
<td>CONVERSIONS</td>
<td>The number of users who opened the link to subscribe to your product after they used the ref tag to navigate to the product page.</td>
</tr>
<tr>
<td>USAGE_HOURS</td>
<td>The amount of usage that is associated with the ref tag.</td>
</tr>
<tr>
<td>REVENUE</td>
<td>The estimated revenue from the associated usage. It's an estimate because customer billing is finalized at the end of each month.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Columns</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE</td>
<td>The name of the product.</td>
</tr>
<tr>
<td>PRODUCT_CODE</td>
<td>A unique identifier for the product that is associated with billing. It's also available in Amazon EC2 instance metadata.</td>
</tr>
<tr>
<td>REFTAG</td>
<td>The name of the ref tag.</td>
</tr>
<tr>
<td>CLICKS</td>
<td>The number of visits to your page with date, ref tag, and Amazon Standard Identification Number (ASIN) grouping.</td>
</tr>
<tr>
<td>CONVERSIONS</td>
<td>The number of users who opened the link to subscribe to your product after they used the ref tag to navigate to the product page.</td>
</tr>
</tbody>
</table>
Data feeds

AWS Marketplace provides a number of data feeds to help sellers collect and analyze information about your product sales. Data feeds are available to all registered AWS Marketplace sellers. Since data feeds are generated within a day, they contain the most current data available.

This page provides an overview of data feeds, and explains how to access and use them. Subsequent pages describe each data feed.

Storage and structure of data feeds

Data feeds collect and deliver comma-separated value (CSV) files to an encrypted Amazon S3 bucket that you provide. The CSV files have the following characteristics:

- They follow 4180 standards.
- Character encoding is UTF-8 without BOM.
- Commas are used as separators between values.
- Fields are escaped by double quotation marks.
- \n is the line feed character.
- Dates are reported in the UTC time zone, are in ISO 8601 date and time format, and are accurate within 1 second.
- All _period_start_date and _period_end_date values are inclusive, which means that 23:59:59 is the last possible timestamp for any day.
- All monetary fields are preceded with a currency field.
- Monetary fields use a period (.) character as a decimal separator, and don't use a comma (,) as a thousands separator.

Data feeds are generated and stored as follows:

- Data feeds are generated within a day, and contain 24 hours of data from the previous day.
- In the Amazon S3 bucket, data feeds are organized by month using the following format:

  bucket-name/data-feed-name_version/year=YYYY/month=MM/data.csv

- As each daily data feed is generated, it is appended to the existing CSV file for that month. When a new month starts, a new CSV file is generated for each data feed.
- Information in data feeds is backfilled from 2010/01/01 to 2020/04/30 (inclusive) and is available in the CSV file (p. 181) in the year=2010/month=01 subfolder.

You may notice cases where the current month's file for a given data feed contains only only column headers, and no data. This means that there were no new entries for that month for the feed. This can happen with data feeds that are updated less frequently, like the product feed. In these cases, data is available in the backfilled folder.

- In Amazon S3, you can create an Amazon S3 lifecycle policy to manage how long to keep files in the bucket.
- You can configure Amazon SNS to notify you when data is delivered to your encrypted S3 bucket. For information on how to configure notifications, see Getting started with Amazon SNS in the Amazon Simple Notification Service Developer Guide.
Historization of the data

Each data feed includes columns that document the history of the data. Except for `valid_to`, these columns are common to all data feeds. They're included as a common history schema and are useful in querying the data.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>valid_from</td>
<td>The first date that the value for the primary key is valid for in relation to values for other fields.</td>
</tr>
<tr>
<td>valid_to</td>
<td>This column is only shown on the Address (p. 184) data feed and is always blank.</td>
</tr>
<tr>
<td>insert_date</td>
<td>The date a record was inserted into the data feed.</td>
</tr>
<tr>
<td>update_date</td>
<td>The date the record was last updated.</td>
</tr>
<tr>
<td>delete_date</td>
<td>This column is always blank.</td>
</tr>
</tbody>
</table>

The following shows an example of these columns.

<table>
<thead>
<tr>
<th>valid_from</th>
<th>valid_to</th>
<th>insert_date</th>
<th>update_date</th>
<th>delete_date</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-12-12T02:32:00Z</td>
<td>2019-03-29T02:00:00Z</td>
<td>2019-03-29T02:00:00Z</td>
<td>2019-03-29T02:00:00Z</td>
<td>2019-03-29T02:00:00Z</td>
</tr>
<tr>
<td>2018-12-20T02:32:00Z</td>
<td>2019-03-29T02:00:00Z</td>
<td>2019-03-29T02:00:00Z</td>
<td>2019-03-29T02:00:00Z</td>
<td>2019-03-29T02:00:00Z</td>
</tr>
<tr>
<td>2019-01-12T02:32:00Z</td>
<td>2019-03-28T03:00:00Z</td>
<td>2019-03-28T03:00:00Z</td>
<td>2019-03-28T03:00:00Z</td>
<td>2019-03-28T03:00:00Z</td>
</tr>
</tbody>
</table>

Accessing data feeds

To access data feeds, you need to configure your environment to receive data feeds to an encrypted Amazon S3 bucket. AWS Marketplace provides an AWS CloudFormation template that you can use to simplify configuration.

**Note**

To access data feeds, you need an IAM role that grants access to AWS Marketplace. You'll use that role as you complete the AWS CloudFormation template. If you don't already have an IAM role, see IAM roles in the IAM User Guide.

To use the AWS CloudFormation template to configure your environment to receive data feeds

1. Go to Set up customer data storage.
2. Choose Create resources with AWS CloudFormation template to open the template in the AWS CloudFormation console in another window.
3. In the template, specify the following and then choose Next:
   - Stack name – The collection of resources you're creating to enable access to data feeds.
   - Amazon S3 bucket name – The bucket for storing data feeds.
   - (Optional) Amazon SNS topic name – The topic for receiving notifications when AWS delivers new data to the Amazon S3 bucket.
4. On the Review page, confirm your entries and choose Create stack.
5. On the next screen, choose the IAM role you created to use with AWS Marketplace (see the 
Note (p. 182) the precedes this procedure) and then choose Next.

6. From the Resources tab, copy Amazon Resource Names (ARNs) for the following resources into the fields on the AWS Marketplace Set up customer data storage page:
   - Amazon S3 bucket for storing data feeds
   - AWS KMS key for encrypting the Amazon S3 bucket
   - (Optional) Amazon SNS topic for receiving notifications when AWS delivers new data to the Amazon S3 bucket

7. On the Set up customer data storage page, choose Submit.

You are now subscribed to data feeds. The next time data feeds are generated, you can access the data.

For more information about AWS CloudFormation templates, see Working with AWS CloudFormation templates in the AWS CloudFormation User Guide.

Using data feeds

When data is available in your Amazon S3 bucket, you can use data feeds in the following ways:

- Download the .CSV files from the Amazon S3 bucket you created in Accessing data feeds (p. 182) so that you can view the data in a spreadsheet.
- Use ETL (extract, transform, and load), SQL query, business analytics tools to collect and analyze the data.

You can use AWS services to collect and analyze data, or any third-party tool that can perform analysis of .CSV-based datasets.

Example: Use AWS services to collect and analyze data

The following procedure assumes that you've already configured your environment to receive data feeds to an Amazon S3 bucket and that the bucket contains data feeds.

To collect and analyze data from data feeds

1. From the AWS Glue console, create a crawler to connect to the Amazon S3 bucket that stores the data feeds, extract the data you want, and create metadata tables in the AWS Glue Data Catalog.

   For more information about AWS Glue, see the AWS Glue Developer Guide.

2. From the Athena console, run SQL queries on the data in the AWS Glue Data Catalog.

   For more information about Athena see the Amazon Athena User Guide.

3. From the Amazon QuickSight console, create an analysis and then create a visual of the data.

   For more information about Amazon QuickSight, see the Amazon QuickSight User Guide.

For a detailed example of one way to use AWS services to collect and analyze data in data feeds, see Using Seller Data Feed Delivery Service, Amazon Athena, and Amazon QuickSight to create seller reports at the AWS Marketplace Blog.

Account data feed

This data feed provides information about all the accounts you interact with: your own, any channel partners you work with, buyers, payers, and all taxed accounts.
Account data is immutable, and it is not associated with a version number. Changes to fields are appended, so this data feed may have several rows with the same account_id and different valid_from values. For information about data history fields, see Historization of the data (p. 182).

The account data feed is refreshed every 24 hours, so new data is available daily.

The following table explains the names and descriptions of the data feed's columns.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account_id</td>
<td>The globally unique identifier (GUID) of the account.</td>
</tr>
<tr>
<td>aws_account_id</td>
<td>The AWS account number of the seller’s AWS account, which is unique by AWS partition.</td>
</tr>
<tr>
<td>encrypted_account_id</td>
<td>The unique, encrypted ID for an individual buyer of your application. The value for encrypted_account_id is used by the AWS Marketplace Metering Service, for example, as the value for CustomerIdentifier that is returned by the ResolveCustomer action.</td>
</tr>
<tr>
<td>mailing_address_id</td>
<td>The mailing address reference for this account.</td>
</tr>
<tr>
<td>tax_address_id</td>
<td>The tax address reference for this account.</td>
</tr>
<tr>
<td>tax_registration_number</td>
<td>For non-US accounts, the tax registration number for this account.</td>
</tr>
<tr>
<td>tax_legal_name</td>
<td>For non-US accounts, the legal company name. This is the name used on tax invoices.</td>
</tr>
</tbody>
</table>

Example of account data feed

The following shows an example of the account data feed. For readability, the data history columns aren't shown. For information about data history fields, see Historization of the data (p. 182).

<table>
<thead>
<tr>
<th>account_id</th>
<th>aws_account_id</th>
<th>encrypted_account_id</th>
<th>mailing_address_id</th>
<th>tax_address_id</th>
<th>tax_registration_number</th>
<th>tax_legal_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>xk0CSmiAm6PQ4Q46e66600T7oMrte6Wv1562</td>
<td>444456660000</td>
<td>Zf7oMzheGWpH</td>
<td>25o3k46eN6eViOlfiiqtxwX8e3kaOiPalUiofyFa3</td>
<td>555567679999</td>
<td>373vuQUqmQ8v</td>
<td>5oJ6vTjSzMrrF2gvh2Vj9HfqiM800MuLEHmyFY5Lr42s8</td>
</tr>
<tr>
<td>7nyo5jwTrOplyQ8vev3TjzmrR5sHmyYogwy0WMhndGU4AfMggmuoTC2j7Pm8ZKKN</td>
<td>737399988888</td>
<td>8SPxAYmi8MwX</td>
<td>NLUc5UeiMlGFTrDWCoftDPhDUF1oaSd8xgl5QM8Db7</td>
<td>50J6vTjSzMrrF2gvh2Vj9HfqiM800MuLEHmyFY5Lr42s8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Address data feed

This data feed provides contact information for all the accounts you interact with: your own, any channel partners you work with, buyers, payers, and all taxed accounts. Each time a new transaction occurs, the customer address for the transaction is scanned, and if it's not in your data feed, a new entry is added to your data feed file.

Address data is immutable.

The address data feed is refreshed every 24 hours, so new data is available daily.
The following table explains the names and descriptions of the data feed's columns.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address_id</td>
<td>The unique key of the address.</td>
</tr>
<tr>
<td>aws_account_id</td>
<td>The AWS account number of this address.</td>
</tr>
<tr>
<td>email_domain</td>
<td>The domain for the email address on file for this account.</td>
</tr>
<tr>
<td>company_name</td>
<td>The company name on file for this account.</td>
</tr>
<tr>
<td>country</td>
<td>The ISO 3166 alpha-2 country code on file for this address.</td>
</tr>
<tr>
<td>state_or_region</td>
<td>The state or region on file for this address.</td>
</tr>
<tr>
<td>city</td>
<td>The city on file for this address.</td>
</tr>
<tr>
<td>postal_code</td>
<td>The postal code on file for this address.</td>
</tr>
<tr>
<td>address_line_1</td>
<td>The first line of the address on file for this address.</td>
</tr>
<tr>
<td>address_line_2</td>
<td>The second line of the address on file for this address.</td>
</tr>
<tr>
<td>address_line_3</td>
<td>The third line of the address on file for this address.</td>
</tr>
</tbody>
</table>

**Example of address data feed**

The following shows an example of the address data feed. In the data feed, this information is presented in a single table. For readability, the data is shown in two tables here, and the data history columns aren't shown. For information about data history fields, see Historization of the data (p. 182).

<table>
<thead>
<tr>
<th>address_id</th>
<th>aws_account_id</th>
<th>email_domain</th>
<th>company_name</th>
<th>country</th>
<th>state_or_region</th>
<th>city</th>
<th>postal_code</th>
</tr>
</thead>
<tbody>
<tr>
<td>V5NhBYBiYogwy0WMhndGU4AfMggmuoTC2j7Pm8ZKKNNyT</td>
<td>444456660000</td>
<td>a.com</td>
<td>Mateo Jackson's Company</td>
<td>DE</td>
<td>Hamburg</td>
<td>67568</td>
<td></td>
</tr>
<tr>
<td>G68xdbkZQDQS9696SYYy6555a86GN895ym</td>
<td>555567679999</td>
<td>b.com</td>
<td>Mary Major's Company</td>
<td>US</td>
<td>OH</td>
<td>57684</td>
<td></td>
</tr>
<tr>
<td>NLUc5UeiMlGFTrDWCoftDPhDUF1oaSd8xgl5QM8Db7</td>
<td>555567679999</td>
<td>c.com</td>
<td>Our Seller</td>
<td>US</td>
<td>NY</td>
<td>89475</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>address_line_1</th>
<th>address_line_2</th>
<th>address_line_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>19th Floor</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Billing event data feed

This data feed provides information about billing events, including invoicing and disbursements. For example, you can use this data feed to learn when and what a buyer is invoiced. You can also use the example SQL queries (p. 191) to analyze the data from this data feed.

This data feed contains information associated with billing events for which you are the seller of record. For agreements made via channel partners, this data feed contains information about billing events between the manufacturer and seller of record.

The billing event data feed is refreshed every 24 hours, so new data is available daily.

Billing event data is immutable.

The following table explains the names and descriptions of the data feed's columns.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>billing_event_id</td>
<td>An identifier for a billing event. This ID is unique in the seller's environment.</td>
</tr>
<tr>
<td>from_account_id</td>
<td>The account that initiated the billing event. If transaction_type is SELLER_REV_SHARE, it is the buyer's payer account. This is a foreign key to the account (p. 183) data feed.</td>
</tr>
<tr>
<td>to_account_id</td>
<td>The account that receives the transaction amount for the product. This is a foreign key to the account data feed.</td>
</tr>
<tr>
<td>end_user_account_id</td>
<td>The account that uses the product. This account may be different from the buyer and payer accounts.</td>
</tr>
<tr>
<td>product_id</td>
<td>The identifier of the product. This is a foreign key to the product (p. 197) data feed.</td>
</tr>
<tr>
<td>action</td>
<td>The type of action for this event. Possible values are as follows:</td>
</tr>
<tr>
<td></td>
<td>• INVOICED – The buyer was invoiced for the amount.</td>
</tr>
<tr>
<td></td>
<td>• FORGIVEN – The buyer was invoiced for the amount, and AWS reverted the charge.</td>
</tr>
<tr>
<td></td>
<td>• DISBURSED – The seller was paid this amount. This can include a month of invoices, or be an on-demand disbursement.</td>
</tr>
<tr>
<td>transaction_type</td>
<td>The type of transaction. For examples, see Taxing scenarios (p. 188). Possible values are as follows:</td>
</tr>
<tr>
<td></td>
<td>• SELLER_REV_SHARE – A positive amount, this is the price the seller set in the agreement with buyer.</td>
</tr>
<tr>
<td></td>
<td>• SELLER_TAX_SHARE – A positive amount, this is the amount added to SELLER_REV_SHARE to cover taxes the seller owes.</td>
</tr>
<tr>
<td>Column name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• AWS_REV_SHARE – A negative amount, this is the listing fee.</td>
<td></td>
</tr>
<tr>
<td>• AWS_TAX_SHARE – A positive amount, this is the amount of taxes AWS collected in addition to SELLER_REV_SHARE. This amount doesn't affect the seller's balance. This amount is not disbursed and is provided for seller's awareness of taxes invoiced to buyer and remitted to authorities on the seller's behalf.</td>
<td></td>
</tr>
<tr>
<td>• transaction_type_REFUND – The amount of refund requested by buyer.</td>
<td></td>
</tr>
<tr>
<td>• transaction_type_CREDIT – The amount AWS credits the buyer.</td>
<td></td>
</tr>
<tr>
<td>• BALANCE_ADJUSTMENT – An adjustment made by AWS to resolve invoicing issues.</td>
<td></td>
</tr>
<tr>
<td>• DISBURSEMENT – If action = DISBURSEMENT and balancing_impacting = 1, this is the amount paid to seller. If the value for action is = INVOICED, this record negates the parent_billing_event_id record either in full or in part. In this case, the related disbursement is disbursement_billing_event_id is shown and balancing_impacting = 0.</td>
<td></td>
</tr>
<tr>
<td>• DISBURSEMENT_FAILURE – Negates the transaction.</td>
<td></td>
</tr>
<tr>
<td>parent_billing_event_id</td>
<td>If the action is DISBURSEMENT or FORGIVEN and the transaction_type is DISBURSEMENT, this is the billing_event_id that initiated this billing event. If action has another value, this field is null.</td>
</tr>
<tr>
<td>disbursement_billing_event_id</td>
<td>The related disbursement when the action is DISBURSED AND one of the following is true:</td>
</tr>
<tr>
<td>• transaction_type like ('SELLER%')</td>
<td></td>
</tr>
<tr>
<td>• transaction_type like ('AWS%')</td>
<td></td>
</tr>
<tr>
<td>In all other cases, this value is null.</td>
<td></td>
</tr>
<tr>
<td>amount</td>
<td>The billing event amount.</td>
</tr>
<tr>
<td>currency</td>
<td>The ISO 639 currency code.</td>
</tr>
<tr>
<td>balance_impacting</td>
<td>Whether the amount is taken into account in calculating seller disbursements. A value of 0 indicates the amount is shown for informational purposes and has no effect on the balance. A value of 1 indicates that this amount takes into account in determining seller disbursements.</td>
</tr>
<tr>
<td>invoice_date</td>
<td>The date the invoice was created.</td>
</tr>
</tbody>
</table>
### Billing event data feed

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>payment_due_date</td>
<td>When the action is INVOICED, the due date for the invoice.</td>
</tr>
<tr>
<td>usage_period_start_date</td>
<td>The start date for the period in the record.</td>
</tr>
<tr>
<td>usage_period_end_date</td>
<td>The end date for the period in the record.</td>
</tr>
<tr>
<td>invoice_id</td>
<td>The AWS invoice ID.</td>
</tr>
<tr>
<td>billing_address_id</td>
<td>The payer's billing address reference in the address data feed.</td>
</tr>
<tr>
<td>transaction_reference_id</td>
<td>An identifier that allows you to cross-reference data from the following reports:</td>
</tr>
<tr>
<td></td>
<td>• Disbursement report (p. 163)</td>
</tr>
<tr>
<td></td>
<td>• Monthly billed revenue report (p. 168)</td>
</tr>
<tr>
<td></td>
<td>• Sales compensation report (p. 174)</td>
</tr>
<tr>
<td></td>
<td>• US sales and use tax report (p. 176)</td>
</tr>
</tbody>
</table>

### Taxing scenarios

The taxation model that is in place for the country and state of the buyer and seller dictates how taxes are collected and remitted. Following are the possible scenarios:

- Taxes are collected and remitted by AWS. In these cases, the transaction_type is AWS_TAX_SHARE.
- Taxes are collected by AWS, disbursed to the seller, and remitted by the seller to the tax authorities. In these cases, the transaction_type is SELLER_TAX_SHARE.
- Taxes are not collected by AWS. The seller must calculate the taxes and remit them to the tax authorities. In these cases, AWS Marketplace doesn't perform tax calculations or receive tax information. The seller pays the taxes from the revenue share.

### Examples of billing event data feed

This section shows examples of the billing event data period at the time of invoicing and one month later. Note the following for all tables in this section:

- In data feeds, billing_event_id values are 40-character alphanumeric strings. They're shown here as two-character strings for readability.
- In the data feed, this information is presented in a single table. For readability, the data is shown in multiple tables here, and all columns aren't shown.

For the examples in this section, assume the following:

- Arnav is the buyer.
  - His account ID is 73739998888.
  - He's located in France, which is subject to marketplace facilitator laws. For more information, see Amazon Web Service Tax Help.
  - He purchased prod-o4grxfacxxxx and was invoiced $120.60 for his monthly usage of that product.
  - He paid the invoice within the month.
AWS Marketplace Seller Guide
Billing event data feed

- Jane is the manufacturer.
- Her account ID is 111122223333.
- Paulo is the seller of record.
  - His account ID is 777788889999.
  - He lives in Kansas, which is not subject to market facilitator laws.

**Billing event data feed for seller of record**

As the seller of record, Paulo invoices the buyer, Arnav.

The following tables show the relevant information in Paulo's data feed when he invoices Arnav.

<table>
<thead>
<tr>
<th>billing_event_id</th>
<th>from_account_id</th>
<th>to_account_id</th>
<th>end_user_account_id</th>
<th>product_id</th>
<th>action</th>
<th>transaction_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>I0</td>
<td>73739998888</td>
<td>7778888999973739998888</td>
<td>prod-04grxfacx???</td>
<td>INVOICED</td>
<td>SELLER_REV_SHARE</td>
<td></td>
</tr>
<tr>
<td>I1</td>
<td>73739998888</td>
<td>AWS</td>
<td>73739998888</td>
<td>prod-04grxfacx???</td>
<td>INVOICED</td>
<td>AWS_TAX_SHARE</td>
</tr>
<tr>
<td>I2</td>
<td>777888899991112223333373739998888</td>
<td>prod-04grxfacx???</td>
<td>INVOICED</td>
<td>SELLER_REV_SHARE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I3</td>
<td>77788889999AWS</td>
<td>73739998888</td>
<td>prod-04grxfacx???</td>
<td>INVOICED</td>
<td>AWS_TAX_SHARE</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>parent_billing_event_id</th>
<th>disbursement_billing_event_id</th>
<th>amount</th>
<th>currency</th>
<th>invoice_date</th>
<th>invoice_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>I0</td>
<td>I14</td>
<td>-100</td>
<td>USD</td>
<td>2018-12-31T00:00:00Z</td>
<td>0612716640</td>
</tr>
</tbody>
</table>

The following tables show the relevant information in Paulo's data feed at the end of the month, after Arnav pays the invoice.

<table>
<thead>
<tr>
<th>billing_event_id</th>
<th>from_account_id</th>
<th>to_account_id</th>
<th>end_user_account_id</th>
<th>product_id</th>
<th>action</th>
<th>transaction_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>I10</td>
<td>73739998888</td>
<td>7778888999973739998888</td>
<td>prod-04grxfacx???</td>
<td>INVOICED</td>
<td>SELLER_REV_SHARE</td>
<td></td>
</tr>
<tr>
<td>I12</td>
<td>777888899991112223333373739998888</td>
<td>prod-04grxfacx???</td>
<td>INVOICED</td>
<td>SELLER_REV_SHARE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I13</td>
<td>77788889999AWS</td>
<td>73739998888</td>
<td>prod-04grxfacx???</td>
<td>INVOICED</td>
<td>AWS_REV_SHARE</td>
<td></td>
</tr>
<tr>
<td>I14</td>
<td>AWS</td>
<td>77788889999</td>
<td></td>
<td>DISBURSED</td>
<td>DISBURSEMENT</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>parent_billing_event_id</th>
<th>disbursement_billing_event_id</th>
<th>amount</th>
<th>currency</th>
<th>invoice_date</th>
<th>invoice_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>I0</td>
<td>I14</td>
<td>-100</td>
<td>USD</td>
<td>2018-12-31T00:00:00Z</td>
<td>0612716640</td>
</tr>
</tbody>
</table>
Billing event data feed

The following tables show the relevant information in the Jane's data feed when Paulo invoices Arnav.

<table>
<thead>
<tr>
<th>billing_event_from_account</th>
<th>to_account_id</th>
<th>end_user_account_id</th>
<th>product_id</th>
<th>action</th>
<th>transaction_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>I5</td>
<td>77778888999911122223333</td>
<td>prod-o4grxfacxxxxx</td>
<td>INVOICED</td>
<td>SELLER_REV_SHARE</td>
<td></td>
</tr>
<tr>
<td>I6</td>
<td>77778888999911122223333</td>
<td>prod-o4grxfacxxxxx</td>
<td>INVOICED</td>
<td>SELLER_TAX_SHARE</td>
<td></td>
</tr>
<tr>
<td>I7</td>
<td>777788889999AWS</td>
<td>prod-o4grxfacxxxxx</td>
<td>INVOICED</td>
<td>AWS_REV_SHARE</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>parent_billing_event_id</th>
<th>disbursement_billing_event_id</th>
<th>amount</th>
<th>currency</th>
<th>invoice_date</th>
<th>invoice_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>I2</td>
<td>I14</td>
<td>80</td>
<td>USD</td>
<td>2018-12-31T00:04:07Z</td>
<td>788576665</td>
</tr>
<tr>
<td>I3</td>
<td>I14</td>
<td>0.2</td>
<td>USD</td>
<td>2018-12-31T00:04:07Z</td>
<td>788576665</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19.8</td>
<td>USD</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following tables show the relevant information in Jane's data feed at the end of the month, after the invoice is paid.

<table>
<thead>
<tr>
<th>billing_event_from_account</th>
<th>to_account_id</th>
<th>end_user_account_id</th>
<th>product_id</th>
<th>action</th>
<th>transaction_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>I30</td>
<td>77778888999911122223333</td>
<td>prod-o4grxfacxxxxx</td>
<td>DISBURSED</td>
<td>SELLER_REV_SHARE</td>
<td></td>
</tr>
<tr>
<td>I31</td>
<td>77778888999911122223333</td>
<td>prod-o4grxfacxxxxx</td>
<td>DISBURSED</td>
<td>SELLER_TAX_SHARE</td>
<td></td>
</tr>
<tr>
<td>I32</td>
<td>777788889999AWS</td>
<td>prod-o4grxfacxxxxx</td>
<td>DISBURSED</td>
<td>AWS_REV_SHARE</td>
<td></td>
</tr>
<tr>
<td>I33</td>
<td>AWS</td>
<td>777788889999</td>
<td>DISBURSED</td>
<td>DISBURSEMENT</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>parent_billing_event_id</th>
<th>disbursement_billing_event_id</th>
<th>amount</th>
<th>currency</th>
<th>invoice_date</th>
<th>invoice_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>I5</td>
<td>I33</td>
<td>-73.5</td>
<td>USD</td>
<td>2018-12-31T00:04:07Z</td>
<td>788576665</td>
</tr>
<tr>
<td>I6</td>
<td>I33</td>
<td>-6.5</td>
<td>USD</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example queries

As described in Using data feeds (p. 183), you can use Athena to run queries on the data that's collected and stored as data feeds in your managed Amazon S3 bucket. This section provides some examples of common ways you might do this. All examples assume that a single currency is used.

Example 1: Amount invoiced, including taxes

To find out how much buyers were invoiced, including taxes, you can run a query like the following:

```sql
SELECT sum(amount) FROM billing_event
WHERE
  action = 'INVOICED'
  AND
  (transaction_type in ('SELLER_REV_SHARE', 'SELLER_TAX_SHARE')
    -- to discard SELLER_REV_SHARE from Manufacturer to Channel Partner, aka cost of goods
    AND to_account_id='seller-account-id'
  )
  OR transaction_type= 'AWS_TAX_SHARE'
);
```

Example 2: Amount invoiced to buyers on seller's behalf

To find out how much buyers were invoiced on a seller's behalf, you can run a query like the following:

```sql
SELECT sum(amount) FROM billing_event
WHERE
  action = 'INVOICED'
  AND transaction_type in ('SELLER_REV_SHARE', 'SELLER_TAX_SHARE')
  AND to_account_id='seller-account-id'
;
```

Example 3: Amount AWS can collect on seller's behalf

To find out how much AWS can collect on a seller's behalf, minus any refunds, credits, and forgiven accounts, you can run a query like the following:

```sql
SELECT sum(amount) FROM billing_event
WHERE
  transaction_type like 'SELLER_%' -- what is invoiced on behalf of SELLER, incl. refunds/credits and cost of goods
  AND action in ('INVOICED','FORGIVEN') -- FORGIVEN action records will "negate" related INVOICED
;
```

Example 4: Amount seller can collect

To find out how much sellers can collect, you can run a query like the following. This example removes listing fees and taxes that AWS collects, and adds any exceptional balance adjustments.
```sql
SELECT sum(amount) FROM billing_event
WHERE
  (transaction_type like 'SELLER_%' -- what is invoiced on behalf of SELLER
   or transaction_type like 'AWS_REV_%' -- what is owed to AWS
   or transaction_type = 'BALANCE_ADJUSTMENT' -- exceptionnal case
  )
  and action in ('INVOICED','FORGIVEN')
;
```

You can also use the following query to collect the same information:

```sql
SELECT sum(amount) FROM billing_event
WHERE
  balance_impacting = 1
  and action in ('INVOICED','FORGIVEN')
;
```

The following example shows the same information, but is restricted to 2018 transactions and assumes all buyers paid their invoices:

```sql
SELECT sum(amount) FROM billing_event
WHERE
  invoice_date between '2018-01-01' and '2018-12-31'
  and balance_impacting = 1
  and action in ('INVOICED','FORGIVEN')
;
```

**Example 5: Amount of disbursements**

To find out the amount that's been disbursed, you can run a query like the following:

```sql
select sum(amount) FROM billing_event
WHERE
  action = 'DISBURSED'
  and transaction_type like 'DISBURSEMENT%'
;
```

**Example 6: Amount pending disbursement**

To find out the amount that's pending disbursement, you can run a query like the following. This query removes amounts that have already been disbursed.

```sql
SELECT sum(amount) FROM billing_event targeted
WHERE
  (transaction_type like 'SELLER_%' -- what is invoiced on behalf of SELLER
   or transaction_type like 'AWS_REV_%' -- what is owed to AWS
   or transaction_type = 'BALANCE_ADJUSTMENT' -- exceptionnal case
  )
  -- DISBURSEMENT action records will "negate" 'INVOICED'
  -- but do not take into account failed disbursements
  AND
  (not exists
    (select 1
      from billing_event disbursement
      join billing_event failed_disbursement
      on disbursement.billing_event_id=failed_disbursement.parent_billing_event_id
      where
        disbursement.action = 'DISBURSED'
        and failed_disbursement.action = 'FORGIVEN'
    )
  )
;
```
Another way to get the same information is to run a query like the following to get the seller's balance:

```sql
SELECT sum(amount) FROM billing_event
WHERE
  balance_impacting = 1
;
```

The following query extends our example. It restricts the results to 2018 transactions and returns more details about the transactions.

```sql
select sum(residual_amount_per_transaction)
from
  (SELECT
    max(billed_invoices.amount) invoiced_amount,
    sum(nvl(disbursed_invoices.amount,0)) disbursed_amount,
    -- Exercise left to the reader:
    -- use transaction_type to distinguish listing fee vs seller-owed money
    -- still pending collection
    max(transaction_type) transaction_type,
    max(billed_invoices.amount)
    + sum(nvl(disbursed_invoices.amount,0)) residual_amount_per_transaction
  FROM billing_event billed_invoices
  -- find related disbursements
  left join billing_event disbursed_invoices
    on disbursed_invoices.action='DISBURSED'
    and disbursed_invoices.parent_billing_event_id=billed_invoices.billing_event_id
  WHERE
    billed_invoices.invoice_date between '2018-01-01' and '2018-12-31'
    and billed_invoices.transaction_type like 'SELLER_%' -- invoiced on behalf of SELLER
    and billed_invoices.action in ('INVOICED','FORGIVEN')
    -- do not take into account failed disbursements
    AND not exists
      (select 1 from billing_event failed_disbursement
        where disbursed_invoices.disbursement_billing_event_id =
        failed_disbursement.parent Billing_event_id
      )
  )
GROUP BY billed_invoices.billing_event_id
;
```

**Example 7: Balance of set of invoices**

To learn the sum of a set of invoices, you can run a query like the following:

```sql
SELECT invoice_id, sum(amount) FROM billing_event targeted
WHERE
  -- invoice_id is only not null for invoiced records AND disbursed records linking them to related disbursement -> no need to filter more precisely
  invoice_id in ('XXX','YYY')
  -- filter out failed disbursements
  AND not exists
    (select 1
      from billing_event disbursement
      join billing_event failed_disbursement
        on failed_disbursement.parent Billing_event_id =
        disbursement.billing_event_id
    )
GROUP BY targeted.invoice_id
;
```
This data feed lists how product IDs and offer IDs map to legacy globally unique identifiers (GUIDs). The legacy GUIDs were used in older reports, and the new IDs are used in data feeds and in AWS Marketplace APIs.

This data feed provides information about all products you've created as the seller of record and all products you're authorized to resell.

The legacy mapping data feed is refreshed every 24 hours, so new data is available daily.

The following table explains the names and descriptions of the data feed's columns.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapping_type</td>
<td>Whether this is a product ID or offer ID.</td>
</tr>
<tr>
<td>legacy_id</td>
<td>The legacy ID for this product or offer.</td>
</tr>
<tr>
<td>new_id</td>
<td>The friendly ID for this product or offer. This ID is used as the primary key and with all current API actions.</td>
</tr>
</tbody>
</table>

### Example of legacy mapping data feed

The following shows an example of the legacy mapping data feed. For readability, the data history columns aren't shown. For information about data history fields, see Historization of the data (p. 182).

<table>
<thead>
<tr>
<th>mapping_type</th>
<th>legacy_id</th>
<th>new_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFFER</td>
<td>8a806c74-db6d-403e-9362-bb08f417ff37</td>
<td>offer-dacpxznfwin</td>
</tr>
<tr>
<td>PRODUCT</td>
<td>1368541d-890b-4b6c-9bb9-4a55306ab642</td>
<td>offer-gszhmle5npzip</td>
</tr>
<tr>
<td>OFFER</td>
<td>558d8382-6b3a-4c75-8348-a627b5525f5f</td>
<td>offer-gszhmle5npzip</td>
</tr>
</tbody>
</table>

### Offer data feed

This data feed provides information about all offers you've created as the seller of record. If a single offer has multiple revisions, all revisions are included in the data feed.

When you make an offer revision and the data in an exposed field changes, a new record is created in the data feed for the same primary key (offer_id plus offer_revision), but with a different value for
valid_from field. For more information about the data feed history columns, see Historization of the data (p. 182).

The offer data feed is refreshed every 24 hours, so new data is available daily.

The following table explains the names and descriptions of the data feed’s columns.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>offer_id</td>
<td>The friendly identifier for the offer.</td>
</tr>
<tr>
<td>offer_revision</td>
<td>The offer revision. This field and the offer_id field combine to form the primary key.</td>
</tr>
<tr>
<td>name</td>
<td>The seller-defined name of the offer.</td>
</tr>
<tr>
<td>expiration_date</td>
<td>The date and time that the offer expires.</td>
</tr>
<tr>
<td>opportunity_name</td>
<td>Any opportunity data linked to this offer. If the offer is bound to an AWS opportunity, this field is populated.</td>
</tr>
<tr>
<td>opportunity_description</td>
<td>Any descriptive information linked to this offer. If the offer is bound to an AWS opportunity, this field is populated.</td>
</tr>
</tbody>
</table>

Example of offer data feed

The following shows an example of the offer data feed. For readability, the data history columns aren’t shown. For information about data history fields, see Historization of the data (p. 182).

<table>
<thead>
<tr>
<th>offer_id</th>
<th>offer_revision</th>
<th>name</th>
<th>expiration_date</th>
<th>opportunity_name</th>
<th>opportunity_description</th>
</tr>
</thead>
<tbody>
<tr>
<td>offer-dacpxznflfwin</td>
<td>1</td>
<td>Enterprise Contract Program Offer</td>
<td>9999-01-01T00:00:00Z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>offer-gszhmle5npzip</td>
<td>1</td>
<td>Private offer created by seller</td>
<td>2020-10-31T00:00:00Z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>offer-hmzhyile8nphlp</td>
<td>1</td>
<td>Enterprise Contract Program Offer</td>
<td>9999-01-01T00:00:00Z</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Offer product data feed

One offer can have several products, and one product can be included in different offers. This data feed lists information about the relationships between offers and products.

This data feed provides information about all product offers you’ve created as the seller of record.

When you add or remove a product from an offer, you create an offer revision.

The offer product data feed is refreshed every 24 hours, so new data is available daily.
The following table explains the names and descriptions of the data feed's columns. For information about the data feed history columns, see Historization of the data (p. 182).

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>offer_id</td>
<td>The friendly identifier of this offer.</td>
</tr>
<tr>
<td>offer_revision</td>
<td>Combines with offer_id field to form the foreign key to the offer revision.</td>
</tr>
<tr>
<td>product_id</td>
<td>The friendly identifier of the product, this is the foreign key to the product that this offer exposes.</td>
</tr>
</tbody>
</table>

### Example of Offer product data feed

The following shows an example of the Offer product data feed.

<table>
<thead>
<tr>
<th>offer_id</th>
<th>offer_revision</th>
<th>product_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>offer-dacpxznflfwin</td>
<td>10</td>
<td>prod-o4grxfafxxxxx</td>
</tr>
<tr>
<td>offer-gszhmle5npzip</td>
<td>24</td>
<td>prod-o4grxfafxxxxy</td>
</tr>
</tbody>
</table>

### Offer target data feed

This data feed lists targets of an offer's revision for all offers you've created as the seller of record. If a single offer has multiple revisions, all revisions are included in the data feed.

When you make an offer revision and the data in an exposed field changes, a new record is created in the data feed for the same primary key (offer_id plus offer_revision), but with a different value for valid_from field.

The offer target data feed is refreshed every 24 hours, so new data is available daily.

The following table explains the names and descriptions of the data feed's columns.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>offer_target_id</td>
<td>The primary key of the feed.</td>
</tr>
<tr>
<td>offer_id+offer_revision</td>
<td>The identifier and revision of the offer. These two columns reference the offer that this target relates to.</td>
</tr>
<tr>
<td>target_type</td>
<td>Indicates whether the offer recipient is BuyerAccounts, which indicates a private offer, or ParticipatingPrograms.</td>
</tr>
</tbody>
</table>
## Product data feed

This data feed provides information about all products you’ve created as the seller of record and all products you’re authorized to resell.

Product data is mutable. This means that when you change the value for one of the following fields, a new record is created in the data feed with a different value for `valid_from` field. For more information about the data feed history columns, see Historization of the data (p. 182).

The product data feed is refreshed every 24 hours, so new data is available daily.

The following table explains the names and descriptions of the data feed's columns.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>product_id</td>
<td>The friendly identifier of the product.</td>
</tr>
<tr>
<td>manufacturer_account_id</td>
<td>The identifier of the product owner. This is a foreign key to the Account(p. 183) data feed.</td>
</tr>
</tbody>
</table>
### Column name

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>product_code</td>
<td>The existing entitlement product code used to meter the product. This value is also used to join data with a report, or to reference what is provided in AWS Marketplace Metering Service.</td>
</tr>
<tr>
<td>title</td>
<td>The title of the product.</td>
</tr>
</tbody>
</table>

### Example of product data feed

The following shows an example of the offer target data feed. For readability, the data history columns aren't shown. For information about data history fields, see Historization of the data (p. 182).

<table>
<thead>
<tr>
<th>product_id</th>
<th>manufacturer_account_id</th>
<th>product_code</th>
<th>title</th>
</tr>
</thead>
<tbody>
<tr>
<td>prod-o4grxfafcxxy</td>
<td>555568000000</td>
<td>product_code_1</td>
<td>Product1</td>
</tr>
<tr>
<td>prod-t3grxfacxxy</td>
<td>444457000000</td>
<td>product_code_2</td>
<td>Product2</td>
</tr>
<tr>
<td>prod-x8faxxfacxxy</td>
<td>666678000000</td>
<td>product_code_3</td>
<td>Product3</td>
</tr>
</tbody>
</table>

### Tax item data feed

This data feed provides information on tax calculations for a customer invoice.

There can be multiple line items (line_item_id) for a given product (product_id) of a given customer invoice (invoice_id), one or more for each tax jurisdiction. This happens, for example, with usage-based bills for customers who are using different AWS region rules by different AWS entities (say, the U.S. and Ireland). To learn more about where AWS collects sales tax, VAT, or GST on your sales and remits such taxes to the local tax authorities, in the name of AWS, Inc., see Amazon Web Service Tax Help.

The tax item data feed is refreshed every 24 hours, so new data is available daily.

Tax item data is immutable.

The following table explains the names and descriptions of the data feed's columns. For information about data history columns, see Historization of the data (p. 182).

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tax_item_id</td>
<td>A unique identifier for a tax item record.</td>
</tr>
<tr>
<td>invoice_id</td>
<td>The AWS invoice ID. You can use this value with the value of product_id to find related tax billing events.</td>
</tr>
<tr>
<td>line_item_id</td>
<td>A unique identifier for a customer bill line item. Refund transactions have the same line item ID as their forward tax transactions.</td>
</tr>
<tr>
<td>customer_bill_id</td>
<td>The unique identifier of the customer bill. Buyers can share this identifier with the seller to help identify and resolve tax calculation questions.</td>
</tr>
<tr>
<td><strong>Column name</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>tax_liable_party</td>
<td>Either AWS or Seller. If the seller is the tax liable party, taxes are collected. If AWS is the tax liable party, sales tax is collected and remitted by AWS. For more information, see <a href="https://aws.amazon.com/marketplace/sellers/tax-collection/">AWS Marketplace Sellers &amp; Tax Collection</a>. If no taxes are collected, there is no value shown here. The seller needs to determine whether some taxes were collected for each invoice, as the seller is liable for tax collection.</td>
</tr>
<tr>
<td>transaction_type_code</td>
<td>The type of transaction. Possible values are as follows:</td>
</tr>
<tr>
<td></td>
<td>• AWS – A forward tax transaction</td>
</tr>
<tr>
<td></td>
<td>• REFUND – A full or partial refund</td>
</tr>
<tr>
<td></td>
<td>• TAXONLYREFUND – A tax-only refund</td>
</tr>
<tr>
<td></td>
<td>Refund transactions share the line item ID with their original forward transactions.</td>
</tr>
<tr>
<td>product_id</td>
<td>A foreign key to the product.</td>
</tr>
<tr>
<td>product_tax_code</td>
<td>A standard code to identify the tax properties for a product. Sellers choose the properties when creating or modifying the product.</td>
</tr>
<tr>
<td>invoice_date</td>
<td>The date the invoice was created.</td>
</tr>
<tr>
<td>taxed_customer_account_id</td>
<td>A foreign key to the account entity who is taxed.</td>
</tr>
<tr>
<td>taxed_customer_country</td>
<td>The ISO 3166 alpha 2 country code of the address used for tax calculations.</td>
</tr>
<tr>
<td>taxed_customer_state_or_region</td>
<td>The state, region, or province used for tax calculations.</td>
</tr>
<tr>
<td>taxed_customer_city</td>
<td>The city used for tax calculations.</td>
</tr>
<tr>
<td>taxed_customer_postal_code</td>
<td>The postal code used for tax calculations.</td>
</tr>
<tr>
<td>tax_location_code_taxed_jurisdiction</td>
<td>The vertex geocode that is associated with the taxed location.</td>
</tr>
<tr>
<td>tax_type_code</td>
<td>The type of tax that is applied to the transaction. The possible values are None, Sales, and SellerUse.</td>
</tr>
<tr>
<td>jurisdiction_level</td>
<td>The jurisdiction level of the address that is used for tax location. The possible values are State, County, City, and District.</td>
</tr>
<tr>
<td>taxed_jurisdiction</td>
<td>The name of the tax jurisdiction.</td>
</tr>
<tr>
<td>display_price_taxability_type</td>
<td>Whether the price that buyers see is inclusive or exclusive of taxes. All AWS Marketplace offerings are exclusive of taxes.</td>
</tr>
</tbody>
</table>
### Column name

<table>
<thead>
<tr>
<th>Column name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>taxable_amount</td>
<td>The amount of the transaction that is taxable, at this jurisdiction level.</td>
</tr>
<tr>
<td>nontaxable_amount</td>
<td>The amount of the transaction that is nontaxable, at this jurisdiction level.</td>
</tr>
<tr>
<td>tax_jurisdiction_rate</td>
<td>The tax rate that is applied, at this jurisdiction level.</td>
</tr>
<tr>
<td>tax_amount</td>
<td>The amount of tax that is charged, at this jurisdiction level.</td>
</tr>
<tr>
<td>tax_currency</td>
<td>The ISO 4217 alpha 3 currency code for above amounts.</td>
</tr>
<tr>
<td>tax_calculation_reason_code</td>
<td>Whether the transaction is taxable, not taxable, exempt, or zero-rated, organized by the jurisdiction level.</td>
</tr>
<tr>
<td>date_used_for_tax_calculation</td>
<td>The date that is used for calculating tax on the transaction.</td>
</tr>
<tr>
<td>customer_exemption_certificate_id</td>
<td>The certificate ID of the exemption certificate.</td>
</tr>
<tr>
<td>customer_exemption_certificate_id_domain</td>
<td>The location where the certificate is stored on Amazon systems.</td>
</tr>
<tr>
<td>customer_exemption_certificate_level</td>
<td>The jurisdiction level that supplied the exemption.</td>
</tr>
<tr>
<td>customer_exemption_code</td>
<td>The code that specifies the exemption; for example, RESALE.</td>
</tr>
<tr>
<td>customer_exemption_domain</td>
<td>The Amazon system that is used to capture the customer exemption information, if available.</td>
</tr>
<tr>
<td>transaction_reference_id</td>
<td>An identifier that allows you to cross-reference data from the following reports:</td>
</tr>
<tr>
<td></td>
<td>- Disbursement report (p. 163)</td>
</tr>
<tr>
<td></td>
<td>- Monthly billed revenue report (p. 168)</td>
</tr>
<tr>
<td></td>
<td>- Sales compensation report (p. 174)</td>
</tr>
<tr>
<td></td>
<td>- US sales and use tax report (p. 176)</td>
</tr>
</tbody>
</table>

### Example of tax item data feed

The following shows an example of the tax item data feed. In the data feed, this information is presented in a single table. For readability, the data is shown in multiple tables here, and all columns aren’t shown.
### Tax Item Data Feed

<table>
<thead>
<tr>
<th>tax_item_id</th>
<th>invoice_id</th>
<th>line_item_id</th>
<th>customer_bill_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>gfkjjobxzgw5ygsrgqjtk54g567d_0002</td>
<td>126059288</td>
<td>23456789</td>
<td>57243900000000000000</td>
</tr>
<tr>
<td>wwk1pqvb8ran3geiw8e34d45zmlgk1_0003</td>
<td>00300000000000000000</td>
<td>123000000000000000000</td>
<td></td>
</tr>
<tr>
<td>wwk1pqvb8ran3geiw8e34d45zmlgk1_0003</td>
<td>00300000000000000000</td>
<td>31200000000000000000</td>
<td></td>
</tr>
<tr>
<td>fnohdid8kwgqq9iui2k30spn3ftgwihbe8h75x_0001</td>
<td>229987654</td>
<td>92100000000000000000</td>
<td></td>
</tr>
<tr>
<td>wwk1pqvb8ran3geiw8e34d45zmlgk1_0002</td>
<td>451431024</td>
<td>99300000000000000000</td>
<td></td>
</tr>
<tr>
<td>wwk1pqvb8ran3geiw8e34d45zmlgk1_0002</td>
<td>451431024</td>
<td>31200000000000000000</td>
<td></td>
</tr>
<tr>
<td>fnohdid8kwgqq9iui2k30spn3ftgwihbe8h75x_0001</td>
<td>229987654</td>
<td>92100000000000000000</td>
<td></td>
</tr>
<tr>
<td>fnohdid8kwgqq9iui2k30spn3ftgwihbe8h75x_0001</td>
<td>229987654</td>
<td>63900000000000000000</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tax_liable_party</th>
<th>transaction_type_code</th>
<th>product_id</th>
<th>product_tax_code</th>
<th>invoice_date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seller</td>
<td>AWS</td>
<td>prod-o4grxfacxxxy</td>
<td>AWSMP_SOFTWARE</td>
<td>2018-12-31T00:00:00Z</td>
</tr>
<tr>
<td>Seller</td>
<td>AWS</td>
<td>prod-o4grxfacxxxy</td>
<td>AWSMP_SOFTWARE</td>
<td>2018-12-31T00:00:00Z</td>
</tr>
<tr>
<td>Seller</td>
<td>AWS</td>
<td>prod-t3grxfacxxxy</td>
<td>AWS_REMOTE_ACCESS</td>
<td>2018-12-31T00:00:00Z</td>
</tr>
<tr>
<td>Seller</td>
<td>REFUND</td>
<td>prod-t3grxfacxxxy</td>
<td>AWS_REMOTE_ACCESS</td>
<td>2018-12-31T00:00:00Z</td>
</tr>
<tr>
<td>Seller</td>
<td>AWS</td>
<td>prod-x8faxxfacxxxy</td>
<td>AWS_REMOTE_ACCESS</td>
<td>2018-12-31T00:00:00Z</td>
</tr>
<tr>
<td>Seller</td>
<td>TAXONLYREFUND</td>
<td>prod-x8faxxfacxxxy</td>
<td>AWS_REMOTE_ACCESS</td>
<td>2018-12-31T00:00:00Z</td>
</tr>
<tr>
<td>AWS</td>
<td>AWS</td>
<td>prod-wghj8xfafrhgj</td>
<td>AWS_REMOTE_ACCESS</td>
<td>2018-12-31T00:00:00Z</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>taxed_customer_ac</th>
<th>taxed_customer_co</th>
<th>taxed_customer_st</th>
<th>taxed_customer_cit</th>
<th>taxed_customer_postal_code</th>
</tr>
</thead>
<tbody>
<tr>
<td>VleGa2t9j3MuxioH9sdxnXXCgGCGu8dxocM5</td>
<td>US</td>
<td>GA</td>
<td>MILTON</td>
<td>48573-4839</td>
</tr>
<tr>
<td>VleGa2t9j3MuxioH9sdxnXXCgGCGu8dxocM5</td>
<td>US</td>
<td>GA</td>
<td>MILTON</td>
<td>48573-4839</td>
</tr>
<tr>
<td>7nyo5jwTrOPlyX81vnj04eEwTur01Ff8b1Q8W8</td>
<td>US</td>
<td>NC</td>
<td>DURHAM</td>
<td>27517-4834</td>
</tr>
<tr>
<td>7nyo5jwTrOPlyX81vnj04eEwTur01Ff8b1Q8W8</td>
<td>US</td>
<td>NC</td>
<td>DURHAM</td>
<td>27517-4834</td>
</tr>
<tr>
<td>7nyo5jwTrOPlyX81vnj04eEwTur01Ff8b1Q8W8</td>
<td>US</td>
<td>NC</td>
<td>NOT APPLICABLE</td>
<td>75844-1235</td>
</tr>
<tr>
<td>7nyo5jwTrOPlyX81vnj04eEwTur01Ff8b1Q8W8</td>
<td>US</td>
<td>NC</td>
<td>HOUSTON</td>
<td>75844-1235</td>
</tr>
<tr>
<td>192a0421313e41f06b52962ed7babf7162b1b688</td>
<td>US</td>
<td>CT</td>
<td>NEW HAVEN</td>
<td>06002-2948</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tax_location_</th>
<th>tax_type_code</th>
<th>jurisdiction_l</th>
<th>taxed_jurisdiction</th>
<th>display_price</th>
<th>taxable_amos</th>
<th>nontaxable_amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>460473664</td>
<td>Sales</td>
<td>State</td>
<td>GA</td>
<td>Exclusive</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>66301164</td>
<td>Sales</td>
<td>County</td>
<td>FULTON</td>
<td>Exclusive</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>692938178</td>
<td>SellerUse</td>
<td>State</td>
<td>NC</td>
<td>Exclusive</td>
<td>58.1</td>
<td>523.8</td>
</tr>
</tbody>
</table>

201
## Tax item data feed

<table>
<thead>
<tr>
<th>Tax location</th>
<th>Tax type code</th>
<th>Jurisdiction level</th>
<th>Taxed jurisdiction</th>
<th>Display price</th>
<th>Taxable amount</th>
<th>Nontaxable amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>692938178</td>
<td>SellerUse</td>
<td>State</td>
<td>NC</td>
<td>Exclusive</td>
<td>-58.1</td>
<td>523.8</td>
</tr>
<tr>
<td>356794387</td>
<td>Sales</td>
<td>State</td>
<td>TX</td>
<td>Exclusive</td>
<td>1105.14</td>
<td>0</td>
</tr>
<tr>
<td>528887443</td>
<td>Sales</td>
<td>City</td>
<td>HOUSTON</td>
<td>Exclusive</td>
<td>-36</td>
<td>0</td>
</tr>
<tr>
<td>171248162</td>
<td>Sales</td>
<td>State</td>
<td>CT</td>
<td>Exclusive</td>
<td>0</td>
<td>114.55</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tax jurisdiction</th>
<th>Tax amount</th>
<th>Tax currency</th>
<th>TaxCalculation reason</th>
<th>Date used for tax calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.206</td>
<td>20.6</td>
<td>USD</td>
<td>Taxable</td>
<td>2018-10-31T00:00:00Z</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>USD</td>
<td>NonTaxable</td>
<td>2018-10-31T00:00:00Z</td>
</tr>
<tr>
<td>0.1</td>
<td>5.8</td>
<td>USD</td>
<td>Taxable</td>
<td>2018-07-31T00:00:00Z</td>
</tr>
<tr>
<td>0.1</td>
<td>-5.8</td>
<td>USD</td>
<td>Taxable</td>
<td>2018-07-31T00:00:00Z</td>
</tr>
<tr>
<td>0.06</td>
<td>66.3</td>
<td>USD</td>
<td>Taxable</td>
<td>2018-07-31T00:00:00Z</td>
</tr>
<tr>
<td>0.01</td>
<td>-0.36</td>
<td>USD</td>
<td>NonTaxable</td>
<td>2018-07-31T00:00:00Z</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>USD</td>
<td>Exempt</td>
<td>2019-06-30T00:00:00Z</td>
</tr>
</tbody>
</table>
AWS Marketplace security

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. The effectiveness of our security is regularly tested and verified by third-party auditors as part of the AWS compliance programs. To learn about the compliance programs that apply to AWS Marketplace, see AWS Services in Scope by Compliance Program.

- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You’re also responsible for other factors including the sensitivity of your data, your organization’s requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AWS Marketplace. The following topics show you how to configure AWS Identity and Access Management to manage access to AWS Marketplace in order to meet your security and compliance objectives. You can also learn how to use other AWS services that can help you to monitor and secure your AWS Marketplace resources.

**Note**
To learn about security on AWS Data Exchange for data products, see Security in the AWS Data Exchange User Guide.

**Topics**
- Controlling access to AWS Marketplace Management Portal (p. 203)
- Policies and permissions for AWS Marketplace sellers (p. 207)
- AWS Marketplace Commerce Analytics Service account permissions (p. 209)
- AWS Marketplace Product Support Connection account permissions (p. 209)
- Amazon SQS permissions (p. 210)
- AWS Marketplace metering and entitlement API permissions (p. 211)
- AMI security policies (p. 212)
- Logging AWS Marketplace API calls with AWS CloudTrail (p. 215)

Controlling access to AWS Marketplace Management Portal

AWS Identity and Access Management (IAM) is an AWS service that helps you control access to AWS resources. If you are an IAM administrator, you control who can be authenticated (signed in) and authorized (have permissions) to use AWS Marketplace resources. IAM is an AWS service that you can use with no additional charge.

The recommended way to control who can do what in AWS Marketplace Management Portal is to use IAM to create users and groups. Then you add the users to the groups, and manage the groups. For example, if John should be allowed to view your products, create an IAM user for him and add his IAM user to a group you create for read-only access. You can assign a policy or permissions to the group that
provide read-only permissions. If you have other users that need read-only access, you can add them to the group you created rather than adding permissions to their user account. If John’s role changes and he no longer needs read-only access, you can remove John’s user account from the group.

A *policy* is a document that defines the permissions that apply to a user, group, or role. In turn, the permissions determine what users can do in AWS. A policy typically allows access to specific actions, and can optionally grant that the actions are allowed for specific resources, like Amazon EC2 instances, Amazon S3 buckets, and so on. Policies can also explicitly deny access. A *permission* is a statement within a policy that allows or denies access to a particular resource. You can state any permission like this: “A has permission to do B to C.” For example, Jane (A) has permission to read messages (B) from John’s Amazon Simple Queue Service queue (C). Whenever Jane sends a request to Amazon SQS to use John’s queue, the service checks to see if she has permission. It further checks to see if the request satisfies the conditions John specified in the permission.

**Important**

All of the IAM users that you create authenticate by using their credentials. However, they use the same AWS account. Any change that a user makes can impact the whole account.

AWS Marketplace has permissions defined to control the actions that someone with those permissions can take in AWS Marketplace Management Portal. There are also policies that AWS Marketplace created and manage that combine several permissions. For example, the `aws-marketplace-management:ViewMarketing` permission gives a user access to the **Marketing** tab in AWS Marketplace Management Portal. The `AWSMarketplaceSellerProductsFullAccess` policy gives the user full access to products in the AWS Marketplace Management Portal.

The following resources provide more information about getting started and using IAM.

- Creating Your First IAM Admin User and Group
- IAM Best Practices
- Managing IAM Policies
- Attaching a Policy to an IAM Group
- Identities (Users, Groups, and Roles)
- Controlling Access Using Policies

The following provides some high-level guidance for creating users and groups, and logging in as an IAM user.

## Creating users

To allow people in your company to sign in to the AWS Marketplace Management Portal, create an IAM user for each person who needs access.

**To create IAM users**

1. Sign in to the AWS Management Console and open the IAM console at [https://console.aws.amazon.com/iam/](https://console.aws.amazon.com/iam/).
2. In the navigation pane, choose **Users** and then choose **Create New Users**.
3. In the numbered text boxes, enter a name for each user that you want to create.
4. Clear the **Generate an access key for each user** check box and then choose **Create**.

**To assign a password to each user that you just created**

1. In the list of users, choose the name of a new user.
2. Choose the **Security Credentials** tab and then choose **Manage Password**.
3. Choose an option for either an auto-generated password or a custom password. Optionally, to require the user to choose a new password at the next sign-in, select the box for **Require user to create a new password at next sign-in**. Choose **Apply**.

4. Choose **Download Credentials** to save the user name, password, and account-specific sign-in URL to a comma-separated values (CSV) file on your computer. Then choose **Close**.

**Note**

To sign in with the IAM user name and password that you just created, users must navigate to your account-specific sign-in URL. This URL is in the credentials file that you just downloaded and is also available on the IAM console. For more information, see How IAM Users Sign In to Your AWS Account in the **IAM User Guide**.

**Tip**

Create a user name and password for yourself as well, even though you're the AWS account owner. It's a recommended best practice for everyone to work in the AWS Marketplace Management Portal as an IAM user, even the account owner. To learn how to create an IAM user for yourself that has administrative permissions, go to Creating an Administrators Group in the **IAM User Guide**.

---

**Creating or using groups**

After you create users, create groups, create permissions to access the pages in the AWS Marketplace Management Portal, add those permissions to the groups, and then add users to the groups.

When you assign permissions to a group, you allow any member of that group to perform specific actions. When you add a new user to the group, that user automatically gains the permissions that are assigned to the group. A group can have permissions for more than one action. We recommend using a managed policy rather than creating your own policy.

**To assign a managed policy for AWS Marketplace to a group**

2. In the navigation pane, choose **Groups**, and then choose the group you want to attach a policy to.
3. On the summary page for the group, under the **Permissions** tab, choose **Attach Policy**.
4. On the **Attach Policy** page, next to **Filter:** enter **awsmarketplace**.
5. Choose the policy or policies you want to attach, and then choose **Attach Policy**.

**To create a policy with AWS Marketplace Management Portal permissions**

2. In the navigation pane, choose **Policies** and then choose **Create Policy**.
3. Next to **Policy Generator**, choose **Select**.
4. On the **Edit Permissions** page, do the following:
   a. For **Effect**, choose **Allow**.
   b. For **AWS Service**, choose **AWS Marketplace Management Portal**.
   c. For **Actions**, select the permission or permissions to allow.
   d. Choose **Add Statement**.
   e. Choose **Next Step**.
5. On the **Review Policy** page, do the following:
   a. For **Policy Name**, enter a name for this policy. Take note of the policy name because you need it for a later step.
To create an IAM group with appropriate permissions and add users to the group

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Groups and then choose Create New Group.
3. For Group Name, type a name for the group. Then choose Next Step.
4. On the Attach Policy page, do the following:
   a. For Filter, choose Customer Managed Policies.
   b. Select the check box next to the name of the policy that you want to attach to this group. This is typically the policy that you just created.
   c. Choose Next Step.
5. Choose Create Group.
6. Find your new group in the Groups list and then select the check box next to it. Choose Group Actions and then Add Users to Group.
7. Select the check box next to each user to add to the group and then choose Add Users.

Signing in as an IAM user

After you have created users in IAM, users can sign in with their own user names and passwords. To do so, they need to use the unique URL that is associated with your AWS account. You can get and distribute the sign-in URL to your users.

To get your account's unique sign-in URL

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Dashboard.
3. Near the top of the content pane, find IAM users sign-in link: and take note of the sign-in link, which has a format like this:

   https://AWS_account_ID.signin.aws.amazon.com/console/

   Note
   If you want the URL for your sign-in page to contain your company name (or other friendly identifier) instead of your AWS account ID, you can create an alias for your account by choosing Customize. For more information, see Your AWS Account ID and Its Alias in the IAM User Guide.
4. Distribute this URL to the people at your company who can work with the AWS Marketplace Management Portal, along with the user name and password that you created for each. Instruct them to use your account's unique sign-in URL to sign in before they access the AWS Marketplace Management Portal.
Policies and permissions for AWS Marketplace sellers

AWS Marketplace has three managed policies you can use with the AWS Marketplace Management Portal. In addition, you can use individual permissions to create your own AWS Identity and Access Management (IAM) policy.

**Note**

To learn about policies and permissions on AWS Data Exchange for data products, see Identity and Access Management in AWS Data Exchange in the *AWS Data Exchange User Guide*.

### Policies for AWS Marketplace sellers

You can use the following managed policies to provide IAM users with controlled access to the AWS Marketplace Management Portal:

**AWSMarketplaceSellerFullAccess**

Allows full access to all of the pages in the AWS Marketplace Management Portal and other AWS services, such as Amazon Machine Image (AMI) management.

**AWSMarketplaceSellerProductsFullAccess**

Allows full access to the Products pages in the AWS Marketplace Management Portal.

**AWSMarketplaceSellerProductsReadOnly**

Allows read-only access to the Products pages in the AWS Marketplace Management Portal.

**Important**

AWS Marketplace buyers can use managed policies to manage the subscriptions they purchase. The managed policies you use with AWS Marketplace Management Portal start with AWSMarketplaceSeller. When you search for policies in IAM, make sure to search for policies that start with AWSMarketplaceSeller.

### Permissions for AWS Marketplace sellers

You can use the following permissions in IAM policies for the AWS Marketplace Management Portal:

/aws-marketplace-management:viewMarketing

Allows access to the Marketing page in the AWS Marketplace Management Portal.

/aws-marketplace-management:viewSupport

Allows access to the Customer Support Eligibility page in the AWS Marketplace Management Portal.

/aws-marketplace-management:viewReports

 Allows access to the Reports page in the AWS Marketplace Management Portal.

/aws-marketplace-management:uploadFiles

Allows access to the File Upload page in the AWS Marketplace Management Portal.

/aws-marketplace-management:viewSettings

Allows access to the Settings page in the AWS Marketplace Management Portal.
aws-marketplace:SearchAgreements

Allows viewing the high-level list of agreements on the Agreements (p. 50) page, as well as opportunities between ISVs and consulting partners on the Partners (p. 47) page.

aws-marketplace:DescribeAgreement

Allows viewing of high-level agreement details on the Agreements page, as well as opportunities between ISVs and consulting partners on the Partners page.

aws-marketplace:GetAgreementTerms

Allows viewing all agreement term details on the Agreements page, as well as opportunities between ISVs and consulting partners on the Partners page.

Note

To enable a user to access the Manage Products page, you must use either the AWSMarketplaceSellerProductsFullAccess or AWSMarketplaceSellerProductsReadOnly managed permissions.

You can combine the preceding permissions into a single IAM policy to grant the permissions that you want. See the following examples.

Example 1: Permissions to access the Marketing and File Upload pages.

To grant permissions to both the Marketing page and the File Upload page, use a policy similar to the following example.

```json
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "aws-marketplace-management:viewMarketing",
      "aws-marketplace-management:uploadFiles"
    ],
    "Resource": "*"
  }]
}
```

Example 2: Permissions to create upgrades and renewals for private offers

To grant permissions to view and use the Agreements page to create upgrades and renewals for private offers, use a policy similar to the following example.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "aws-marketplace:SearchAgreements",
        "aws-marketplace:DescribeAgreement",
        "aws-marketplace:GetAgreementTerms"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```
"Condition": {
    "StringEquals": {
        "aws-marketplace:PartyType": "Proposer"
    },
    "ForAllValues:StringEquals": {
        "aws-marketplace:AgreementType": ["PurchaseAgreement"
    ]
    }
}

**Using IAM groups**

Alternatively, you can create separate IAM groups for granting access to each individual page in the AWS Marketplace Management Portal. Users can belong to more than one group. So, if a user needs access to more than one page, you can add the user to all of the appropriate groups. For example, create one IAM group and grant that group permission to access the Marketing page, create another group and grant that group permission to access the File Upload page, and so on. If a user needs permission to access both the Marketing page and the File Upload page, add the user to both groups.

For more information about IAM users and groups, see Identities (Users, Groups, and Roles) in the IAM User Guide.

**AWS Marketplace Commerce Analytics Service account permissions**

You can use the following IAM permission policy to allow an IAM user to access the AWS Marketplace Commerce Analytics Service.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "marketplacecommerceanalytics:GenerateDataSet",
            "Resource": "*"
        }
    ]
}
```

For more information about this feature, see AWS Marketplace Commerce Analytics Service (p. 7).

**AWS Marketplace Product Support Connection account permissions**

The AWS Marketplace product support connection feature makes it possible for customers to provide contact information in the AWS Marketplace website so that you can offer them support for your products. AWS Marketplace shares the data that the customer provides to you through an API. Customers can choose to add contact details during or after they purchase a product that you enrolled in AWS
Marketplace product support connection. You use the API to retrieve the customer's contact data, along with relevant product subscription details.

If haven't enrolled in the section called "AWS Marketplace Commerce Analytics Service" (p. 7), you must configure your account and AWS services to use it. Do the following:

1. (Optional) Create an IAM user.
2. Create a destination Amazon Simple Storage Service (Amazon S3) bucket.
3. Create an Amazon Simple Notification Service (Amazon SNS) topic for response notifications.
5. (Recommended) Make a test call to the service using the AWS Command Line Interface (AWS CLI).

For instructions, see the Onboarding guide (p. 9).

Note
The IAM permissions required for product support connection are different from those required for commerce analytics service. Product support connection requires that the IAM user can call the marketplacecommerceanalytics:StartSupportDataExport action.

You can allow an IAM user to call the StartSupportDataExport action by using an IAM permission policy.

Example

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "marketplacecommerceanalytics:StartSupportDataExport",
      "Resource": "*"
    }
  ]
}
```

For more information about this feature, see Product Support Connection (p. 19).

---

**Amazon SQS permissions**

As part of the SaaS product publication process, AWS Marketplace provides you an Amazon SNS topic you can use to receive notifications if a customer's subscription or entitlement status changes. You can configure one or more Amazon SQS queues to the topic so that the queues can take action on the notification. For example, if a customer adds more storage to the subscription they have to your SaaS product, the Amazon SNS topic can send a message to an Amazon SQS queue that starts a process to automatically increase the storage capacity available to that customer.

When you subscribe the Amazon Simple Queue Service (Amazon SQS) queue to the provided Amazon SNS topic, permissions are automatically added to allow the topic to publish messages to the queue. However, you still need an IAM policy for granting the AWS Marketplace Metering and Entitlement Service API user access to the queue. This can be applied to the same user if the services run with the same credentials. Create a policy with the following contents and attach it to your IAM user or role.

```json
{
  "Version": "2012-10-17",
  "Statement": [
```

---

210
AWS Marketplace Seller Guide
AWS Marketplace metering and entitlement API permissions

Software as a service (SaaS) products, AMI products, and container products can use the AWS Marketplace Metering and Entitlement Service API. Each type requires different IAM permissions. For your product or products, you meter for all usage, and customers are billed by AWS based on the metering records that you provide. To enable the integration required to provide AWS Marketplace your metering records, the service account that the integration is running under needs a constrained IAM policy to enable access. Attach the policy for the product type you are sending metering information for to the IAM user or role that you’re using for the integration.

**IAM policy for SaaS products**

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "aws-marketplace:ResolveCustomer",
                "aws-marketplace:BatchMeterUsage",
                "aws-marketplace:GetEntitlements"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

**Note**
The first permission is required for all SaaS integrations. The second and third permissions are needed for the AWS Marketplace metering service API and the AWS Marketplace entitlement service API, respectively.

**IAM policy for AMI products**

```json
{
    "Version": "2012-10-17",
}
```
IAM policy for container products

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "aws-marketplace:RegisterUsage"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

For more information about creating IAM users, see Creating an IAM User in Your AWS Account in the IAM User Guide. For more information about creating and assigning policies, see Changing Permissions for an IAM User.

This policy grants access to the APIs for the IAM role or user that you attach the policy to. For more information on how to enable role assumption by another account for these API calls, see How to Best Architect Your AWS Marketplace SaaS Subscription Across Multiple AWS Accounts at the AWS Partner Network (APN) Blog.

AMI security policies

AWS Marketplace maintains the following policies for all Amazon Machine Image (AMI) products and offerings in AWS Marketplace. The policies promote a safe, secure, and trustworthy platform for our customers.

All products and their related metadata are reviewed when they're submitted to ensure that they meet or exceed current AWS Marketplace policies. These policies are reviewed and adjusted to meet evolving security guidelines. AWS Marketplace continuously scans your products to verify that they meet changes to the security guidelines. If products fall out of compliance, we might require that you update your AMI product to meet new standards. Likewise, if a newly discovered vulnerability is found to affect the AMI, we will ask you to provide an updated AMI with the relevant updates in place. You must use the self-service AMI scanning tool before submitting your AMI. This tool helps ensure that the AMI meets AWS Marketplace policies.

Security policies

All AMIs must adhere to the following security policies:

- AMIs must not contain any known vulnerabilities, malware, or viruses as detected by the self-service AMI scanning tool or AWS Security.
AMIs must use currently supported operating systems and other software packages. Any version of an
AMI with an End-of-Life (EoL) operating system or other software packages will be delisted from the
AWS Marketplace. You can build a new AMI with updated packages and publish it as a new version to
AWS Marketplace.

All instance authentication must use key pair access, not password-based authentication, even if the
password is generated, reset, or defined by the user at launch. AMIs must not contain passwords,
authentication keys, key pairs, security keys, or other credentials for any reason.

AMIs must not request or use access or secret keys from users to access AWS resources. If your AMI
application requires access to the user account, it must be achieved through an AWS Identity and
Access Management (IAM) role instantiated through AWS CloudFormation, which creates the instance
and associates the appropriate role. When single-AMI launch is enabled for products with an AWS
CloudFormation delivery method, corresponding usage instructions must include clear guidance for
creating minimally privileged IAM roles. For more information, see the section called “AMI-based
Delivery Using CloudFormation” (p. 70).

Linux-based AMIs must not allow SSH password authentication. Disable password authentication via
your sshd_config file by setting PasswordAuthentication to NO.

Access policies

There are three categories of access policies: general, Linux-specific, and Windows-specific policies.

General access policies

All AMIs must adhere to the following general access policies:

- AMIs must allow operating system (OS)-level administration capabilities to allow for compliance
requirements, vulnerability updates, and log file access. Linux-based AMIs use SSH, and Windows-
based AMIs use RDP.
- AMIs must not contain authorized passwords or authorized keys.
- AMIs must not use fixed passwords for administrative access. AMIs must use a randomized password
instead. An alternative implementation is to retrieve the instance metadata and use the instance_id
as the password. The administrator must be prompted for this randomized password before being
permitted to set or change their own credentials. For information about retrieving instance metadata,
see Instance Metadata and User Data in the Amazon EC2 User Guide for Linux Instances.
- You must not have access to the customer’s running instances. The customer has to explicitly enable
any outside access, and any accessibility built into the AMI must be off by default.

Linux-specific access policies

Linux-based AMIs must adhere to the following access policies, as well as the general access policies:

- Linux-based AMIs must disable password-based remote logins for root access and allow only sudo
access through a user account, not root. Users must use sudo access through a user account and can’t
use root access. Sudo access allows the administrator to control which users are allowed to perform
root functions. It also logs the activity for an audit trail. AMIs must not contain authorized passwords
or authorized keys.
- Linux-based AMIs must not have blank or null root passwords.

Windows-specific access policies

Windows-based AMIs must adhere to the following access policies, as well as the general access policies:
• For Windows Server 2016 and later, use EC2Launch.
• For Windows Server 2012 R2 and earlier, use the most recent version of Ec2ConfigService and enable Ec2SetPassword, Ec2WindowsActivate, and Ec2HandleUserData.
• Remove guest accounts and remote desktop users, none of which are allowed.

Customer information policies

All AMIs must adhere to the following customer information policies:

• AMI products must not require customers to register with the seller or require customers to provide any identifying information to use the product, except as required by BYOL (Bring Your Own License) products.
• Software must not require, collect, or export customer data without the customer's knowledge and express consent.

Product usage policies

All AMIs must adhere to the following product usage policies:

• Products must not restrict access to the product or product functionality by time, number of users, or other restrictions. Beta and prerelease products, or products whose sole purpose is to offer trial or evaluation functionality, are not supported. Developer, Community, and BYOL editions of commercial software are supported, provided an equivalent paid version is also available in AWS Marketplace.
• All AMIs must be compatible with either the Launch from Website experience or AMI-based delivery through AWS CloudFormation. For Launch from Website, the AMI can't require customer or user data at instance creation to function correctly.
• Each AMI must contain everything that a buyer needs to use the software, including any client applications.
• For all products except BYOL, the fulfillment process must not require the customer to leave AWS Marketplace.
• AMIs must not require a subscription API or launches from outside AWS Marketplace.
• Product software and metadata must not contain language that redirects users to other cloud platforms, additional products, or upsell services that aren't available in AWS Marketplace.

Architecture policies

All AMIs must adhere to the following architecture policies:

• Source AMIs for AWS Marketplace must be provided in the US East (N. Virginia) Region.
• AMIs must use HVM virtualization.
• AMIs must use 64-bit or 64-bit ARM architecture.
• AMIs must be AMIs backed by Amazon Elastic Block Store (Amazon EBS). We don't support AMIs backed by Amazon Simple Storage Service (Amazon S3).
• AMIs must use a supported file system to pass self-service AMI scanning tool validation. The supported file systems are Ext2, Ext3, Ext4, Xfs, Vfat, Lvm, and NTFS. Encrypted file systems aren't supported.
• FreeBSD products must be built from a Linux-based OS.
• AMIs must be built so that they can run in all AWS Regions and are Region-agnostic. AMIs built differently for different Regions aren't allowed.
Logging AWS Marketplace API calls with AWS CloudTrail

AWS Marketplace is integrated with CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Marketplace. CloudTrail captures API calls for AWS Marketplace as events. The calls captured include calls from the AWS Marketplace console and code calls to the AWS Marketplace API operations.

CloudTrail is enabled on your AWS account when you create the account. When supported event activity occurs in AWS Marketplace, that activity is recorded in a CloudTrail event along with other AWS service events in Event history. You can view, search, and download recent events in your account.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or a federated user.
- Whether the request was made by another AWS service.

AWS Marketplace supports logging the BatchMeterUsage operation as events in CloudTrail log files.

**Example: AWS Marketplace Log File Entries**

The following example shows a CloudTrail log entry that demonstrates the BatchMeterUsage action from the AWS Marketplace Metering Service.

```json
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "userName": "Alice"
      },
      "eventTime": "2018-04-19T16:32:51Z",
      "eventSource": "metering-marketplace.amazonaws.com",
      "eventName": "BatchMeterUsage",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.0.2/24",
      "userAgent": "Coral/Netty14",
      "requestParameters": {
        "usageRecords": [
          {
            "dimension": "Dimension1",
            "timestamp": "Apr 19, 2018 4:32:50 PM",
            "customerIdentifier": "customer1",
            "quantity": 1
          }
        ],
        "productCode": "EXAMPLE_productCode"
      },
      "responseElements": {
```
"results": [  
  
  
  
  "usageRecord": {  
    "dimension": "Dimension1",  
    "timestamp": "Apr 19, 2018 4:32:50 PM",  
    "customerIdentifier": "customer1",  
    "quantity": 1  
  
  },  
  "meteringRecordId": "bEXAMPLE-98f0-4e90-8bd2-bf0EXAMPLE1e",  
  "status": "Success"  
  },  
  "unprocessedRecords": [  
  ]  
  },  
  "requestID": "dEXAMPLE-251d-11e7-8d11-1f3EXAMPLE8b",  
  "eventID": "cEXAMPLE-e6c2-465d-b47f-150EXAMPLE97",  
  "readOnly": false,  
  "eventType": "AwsApiCall",  
  "recipientAccountId": "123456789012"  
}  
]  
} 

Related Topics

For more information, see the following topics in the AWS CloudTrail User Guide:

- Overview for Creating a Trail
- AWS Service Integrations with CloudTrail Logs
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts
- CloudTrail userIdentity Element.
# Document history

The following table describes the documentation for this release of the *AWS Marketplace Seller Guide*.

<table>
<thead>
<tr>
<th>update-history-change</th>
<th>update-history-description</th>
<th>update-history-date</th>
</tr>
</thead>
<tbody>
<tr>
<td>More eligible jurisdictions</td>
<td>The following are now eligible to become sellers on AWS Marketplace: Bahrain, Norway, Switzerland, and the United Arab Emirates (UAE).</td>
<td>June 17, 2020</td>
</tr>
<tr>
<td>You can offer upgrades and renewals on accepted private offers</td>
<td>For SaaS contract and SaaS contracts with consumption products, you can offer upgrades and renewals using private offers on previously-accepted private offers.</td>
<td>May 28, 2020</td>
</tr>
<tr>
<td>More information is available in data feeds</td>
<td>More information from reports is broken down into smaller data feeds to simplify finding and analyzing data.</td>
<td>May 21, 2020</td>
</tr>
<tr>
<td>Standardized license terms are now available</td>
<td>You can offer standardized license terms in place of custom EULAs to simplify the contracting process.</td>
<td>April 28, 2020</td>
</tr>
<tr>
<td>Australia and New Zealand are eligible jurisdictions</td>
<td>The following are now eligible to become sellers on AWS Marketplace: (i) Permanent residents and citizens of Australia (AU) New Zealand (NZ) or (ii) business entities organized or incorporated in one of those areas.</td>
<td>April 2, 2020</td>
</tr>
<tr>
<td>Container products now support custom metering and pricing enhancements</td>
<td>If you want to define your own pricing units and meter that usage to us for billing, integrate with the AWS Marketplace Metering Service’s <code>meterUsage</code> action.</td>
<td>November 14, 2019</td>
</tr>
<tr>
<td>AWS Marketplace supports data products through AWS Data Exchange</td>
<td>You can now provide data products in AWS Marketplace.</td>
<td>November 13, 2019</td>
</tr>
<tr>
<td>Introducing the AWS Marketplace Catalog API service (p. 217)</td>
<td>The AWS Marketplace Catalog API service provides an API interface for approved sellers to programmatically manage their products.</td>
<td>November 12, 2019</td>
</tr>
<tr>
<td>AWS Marketplace supports paid hourly containers</td>
<td>AWS Marketplace now supports paid hourly containers running on Amazon Elastic Kubernetes Service (Amazon EKS).</td>
<td>September 25, 2019</td>
</tr>
<tr>
<td>Updated AMI product functionality</td>
<td>You can now deploy AMIs and Lambda functions together using AWS CloudFormation.</td>
<td>September 11, 2019</td>
</tr>
<tr>
<td>Added Security section</td>
<td>Consolidated security content under a new Security section.</td>
<td>May 7, 2019</td>
</tr>
<tr>
<td>Updated AMI security policies</td>
<td>Updated the security policies for AMI products</td>
<td>April 11, 2019</td>
</tr>
<tr>
<td>Added versioning information to Machine Learning Products section (p. 217)</td>
<td>Added content describing product version capability for machine learning products.</td>
<td>March 21, 2019</td>
</tr>
<tr>
<td>Added Machine Learning Products section</td>
<td>Added content for publishing machine learning products</td>
<td>November 28, 2018</td>
</tr>
<tr>
<td>Added Container-Based Products section</td>
<td>Added content for publishing container-based products.</td>
<td>November 27, 2018</td>
</tr>
<tr>
<td>Updated link for submitting seller help request</td>
<td>Changed email address to webform address.</td>
<td>October 22, 2018</td>
</tr>
<tr>
<td>Added SaaS contracts with consumption content</td>
<td>Restructured SaaS content and added content to support release of SaaS contracts with consumption features.</td>
<td>October 18, 2018</td>
</tr>
<tr>
<td>Added content about flexible payment schedule for private offers</td>
<td>Added content to support release of flexible payment scheduler for private offers content.</td>
<td>October 15, 2018</td>
</tr>
<tr>
<td>Updated IAM permissions content</td>
<td>Added content to support of new IAM permission for AMMP read only access.</td>
<td>October 9, 2018</td>
</tr>
<tr>
<td>Added content about consulting partner private offers</td>
<td>Added content to support Consulting Partner Private Offers feature release.</td>
<td>October 9, 2018</td>
</tr>
<tr>
<td>Added content about private image builds</td>
<td>Added content to support release of Private Image Build for AMIs feature.</td>
<td>August 13, 2018</td>
</tr>
<tr>
<td>Added search engine optimization guidance for sellers.</td>
<td>Added guidance for sellers who want to optimize their product for search.</td>
<td>July 3, 2018</td>
</tr>
<tr>
<td>Updated link to AWS Marketplace logos</td>
<td>Updated link to point to new AWS Marketplace logos.</td>
<td>June 12, 2018</td>
</tr>
<tr>
<td>Added seller guides</td>
<td>Converted all PDF seller guides to online content.</td>
<td>May 9, 2018</td>
</tr>
</tbody>
</table>
AWS glossary

For the latest AWS terminology, see the AWS glossary in the AWS General Reference.